

# Microcomputer Products Group

**u note Reprints**

**It took the minicomputer company to make micros this easy.**

## TABLE OF CONTENTS

		PAGE NO.
001	BATTERY BACKUP	1
005	IEEE BUS SUB-SPECS	4
009	HEX AND QUAD HOLD-DOWN BRACKET FOR LSI-11 SYSTEMS	5
010A	POWER SUPPLY FOR H909C	6
011	LSI-11 HALTING DURING INTERRUPT CYCLE	8
012	LSI-11 BUS THEORY OF OPERATION	10
013	INSTALLING APL-11 ON 11V03 AND 11T03	19
014	CORRECT INPUT PARAMETERS FOR THE QJV11 PROM FORMATTING PROGRAM	20
015	POWER SEQUENCING FOR THE KD11-HA MODULE	22
017	LSI-11/2 PROCESSOR CLOCK	25
018	DR11-C vs. DRV11	26
019	EIA RS-422 AND RS-423	30
020	9 X 6 SLOT BACKPLANE DOCUMENTATION ERROR	34
021	COMPARISON OF DATA TRANSMISSION TECHNIQUES	35
023	USING THE MSV11-D 30K OPTION	40
024A	ASYNC., SERIAL LINE UNIT COMPARISONS	41
025	CONFIGURING MEMORY SYSTEMS WITH MSV11-D RAM & PROM	46
026	MICRO BACKPLANE MECHANICAL MOUNTING GUIDELINES	48
027A	PROM CHIPS AVAILABLE UNDER PART # MRV11-AC	56
028	EXTENDED MEMORY FOR THE LSI-11	57
029	USING THE MRV11-AA FOR A BOOTSTRAP ROM	69
030	SRUN SIGNAL	71
032	EXTENDED BUS TIME-OUT LOGIC	73
033	CABLES FOR DLV11, DLV11-E, DLV11-F	76

034	CONFIGURING A 3-Box 11/03 SYSTEM	78
035	PROM PROGRAMMING	80
036	CORE MEMORY IN 11/03-L BACKPLANE	81
037	C-D INTERCONNECT SCHEME	82
038	DIAGNOSTICS FOR 30K MEMORIES ON LSI-11's	85
039	DMA REQUEST/GRANT TIMING	86
040	PATCHES FOR BASIC/PTS ON LSI-11	88
041	NEW FUNCTIONALITY FOR BDV11-AA BOOT	90
042	REMOVING MODULES FROM "LIVE" BACKPLANES	91
043	BACKPLANES FOR THE RLV11 (RL01)	93
044	CONSOLE ODT "L" COMMAND ON 30K SYSTEMS	95
046A	DLV11-F REPLACEMENT FOR THE DLV11	96
047	INCOMPATIBILITY BETWEEN THE REV11 AND THE LSI-11/23	99
048	LSI-11/23 INSTRUCTION TIMING (PRELIMINARY)	101
049	SYSTEM DIFFERENCES - LSI-11 vs. LSI-11/23	111
050	MICRO ODT DIFFERENCES - LSI-11 vs. LSI-11/23	113
051	DIGITAL SUPPORTED PROM's	116
052	PARITY MEMORY IN LSI-11/23 SYSTEMS	117
053	PDP-11 FAMILY DIFFERENCES	125
054	MXV11 CONFIGURATION	136
055	LSI-11 vs. LSI-11/23 BUS TIMING	141
056	DLV11-J CABLING	143
057	LOCATION OF W13 ON THE BDV11	147
058	CONFIGURING MEMORY FOR LSI-11 SYSTEMS WITH MORE THAN 64K BYTES	149
059	LSI-11/23 FOUR-LEVEL INTERRUPTS	155
060	MAXIMUM CONFIGURATION OF DLV11-J MODULES	162

061	PROGRAMMING THE MRV11-C	164
062	BOOTSTRAPS FOR TU58, RL01, RK05, RX02, RX01	173
063	RL01 TYPE-IN BOOTSTRAP	204
064	DLV11-J I/O PAGE ADDRESS PROBLEM REPORT	205
065	BOOTSTRAP FOR RX02	207
066	11/23 FLOATING POINT COMPATIBILITY	210
067A	DLV11-J RECEIVER CHIP PROBLEM	211
068	MICROCOMPUTER MODULE ENVIRONMENTAL CONSIDERATIONS	213
069	18-BIT DMA WITH CHIPKITS	214
070	LSI-11 vs. LSI-11/23 TRANSACTION DIFFERENCES	216
071	EXPANDING BA11-MA AND BA11-NC BASED SYSTEMS	218
072	PERIPHERAL COMPATIBILITY WITH 11/23 SYSTEMS	221
073	TU58 CABLING	223
074	MXV11-AA, -AC CABLING	225
075	MXV11-A2 BOOTSTRAP ERROR HALTS	227
077	SUMMARY OF BOOTSTRAP SOURCES	230
078	LSI-11/23 PROCESSOR DIFFERENCES	231
079	THE LSI-11/23 AND THE LSI-11/2 BUSES ARE THE SAME	233
080A	LSI-11/23 I/O PAGE ADDRESSING	234
081	USE OF RECOMMENDED DISKETTES	236
082	HANDLERS FOR SERIAL LINE PRINTERS	237
083	ALTERNATE CLOCK FREQUENCIES FOR THE MXV11	248
084	IMPROVED DLV11-F	250
085	WAKE-UP CIRCUIT IMPLEMENTATIONS	252

<b>ynote</b>		NUMBER 001
TITLE	<u>BATTERY BACKUP</u>	DATE 4 / 1 /77
DISTRIBUTION	<u>UNRESTRICTED</u>	PRODUCT MSV11-C (M7955)
ORIGINATOR	<u>JOE AUSTIN</u>	PAGE 1 OF 3

### SCOPE

The function of this note is to furnish the information necessary to provide battery backup for the MSV11-C 16K MOS RAM module. Specific recommendations for the battery, charger, and DC-DC converters will not be covered at this time.

Additional information on the MSV11-C is contained in the "MSV11-C Users Manual", document number EK-MSV11-OP.

### SYSTEM DESCRIPTION

A block diagram showing two MSV11-C modules providing 28K words of memory is shown in Figure 2. Each 16K module is plugged into a standard backplane with the battery power connected to pins AV1 (+5V) and AS1 (+12V) as shown. These two voltages are sourced by DC-DC converters/regulators which are connected directly to the battery. A charger is provided which is connected to the battery and converters by control logic. The function of the control logic is to disconnect the charger when AC power is lost, to disconnect the battery from the converters when it discharges too low, and to switch to a trickle-charge once the battery has been fully charged.

### FUNCTIONAL REQUIREMENTS

To properly implement this system, the following conditions must be met:

1. The battery power required to back up the two modules is:

+5V <u>+3%</u>	1.6 A type (2.8A max)
+12V <u>+3%</u>	0.32 A type (0.4A max)

These voltages must remain within +3% of the voltages for the LSI-11 at all times, and must not change by more than +3% during the transition from the battery.

**digital**

**COMPONENTS  
GROUP**

2. All MSV11-C modules using battery backup must be at etch Rev. D. Those modules at etch Rev. C must have ECO No. 1 for module M7955 installed. This ECO adds the circuit shown in Figure 1 in the following manner:
  - cut the etch to free pins E31-5 and E31-10
  - add wires from E18-12 to E20-9  
E19-12 to E20-10  
E20-14 to E31-5 & 10
3. The following jumpers should be configured as shown:
  - W1, W5 - Remove to separate the battery power from the bussed system power.
  - W2, W3 - Insert to connect the battery power to the refresh logic.
  - W6, W7 - Insert to enable internal refresh and to prevent the module from asserting BRPLY during refresh.
4. The heat generated by the refresh logic on each MSV11-C is 8 watts. Alternate cooling must be provided to dissipate this heat if the AC fans are off for more than two minutes.
5. Certain precautions should be taken in the design of the power sequencing logic and other special modules. Signal BSYNC must be driven by a source that has a high impedance when power is removed from the rest of the system. Signals BDCOK and BPOK must be driven by a low impedance source (less than 8 ohms) under the same condition and should have no bounce (no relay). A J-FET is recommended since it also provides the necessary rise and fall times. These precautions have been taken in those systems using the H780-H or J power supply as provided by PDP-11/03 or PDP-11V03 systems.
6. All master (DMA) modules must finish gracefully when BPOK indicates an AC power failure. Any cycles in process must be allowed to finish, should be off the bus within two microseconds, and must not hang up signal BSYNC.
7. The software must finish its power-down procedure and issue a HALT instruction within two milliseconds from the time signal BPOK indicates an AC power failure.

(E18-12) DCOK L

(E19-12) Lockout H

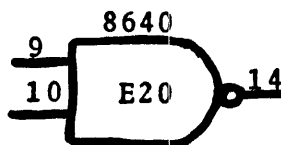
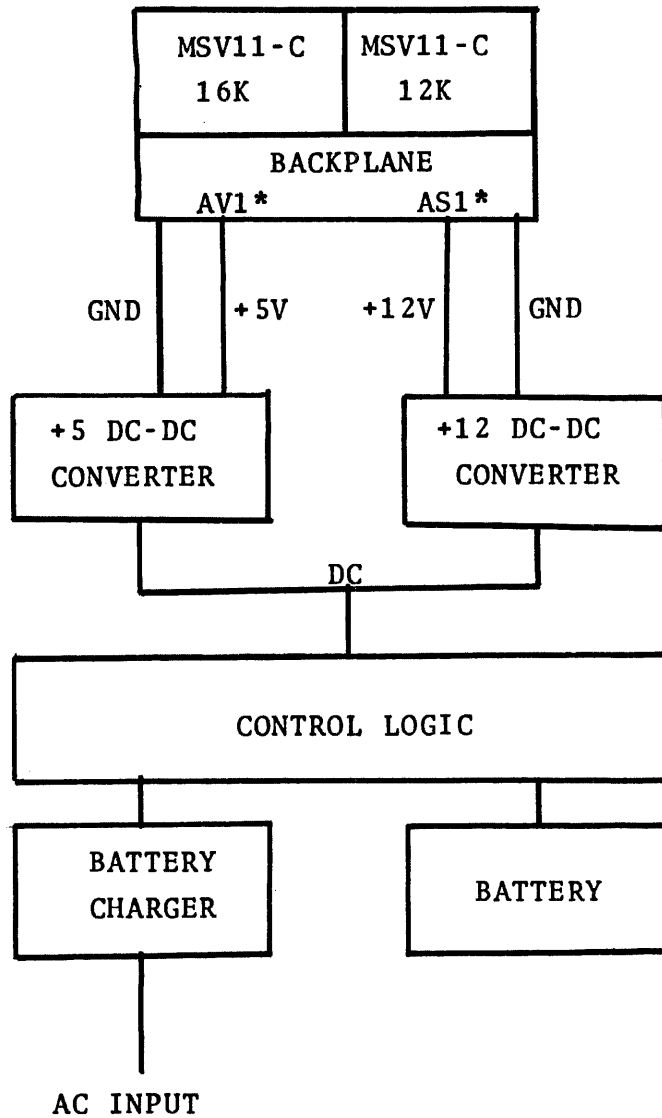


FIGURE 1. ECO LOGIC

digital

COMPONENTS  
GROUP

**FIGURE 2. TYPICAL BATTERY BACKUP SYSTEM**



<b>μnote</b>		NUMBER
		005
TITLE	IEEE BUS SUB-SPECS	DATE
		5 / 2 / 77
DISTRIBUTION	IBV11 CUSTOMERS	PRODUCT
		IBV11-A (M7954)
ORIGINATOR	JOE AUSTIN	PAGE OF
		1 1


The IBV11-A, when connected to the LSI-11, will meet the following subsets of IEEE Standard 488-1975:

SH1	SR1	C1
AH1	RL1	C2
T5	PP2	C3
TE5	DC1	C4
L3	DT1	C5
LE3		

This module is designed to be the only controller on the IEEE bus. Therefore, it will not respond to another controller on the bus that issues either a parallel poll configure command or a parallel poll control signal (subset PP2).

**digital**  
**COMPONENTS**  
**GROUP**



	NUMBER 009
	DATE 6 / 6 / 77
	PRODUCT LSI-11 MOUNTING HARDWARE
	PAGE 1 OF 1
TITLE HEX AND QUAD HOLD-DOWN BRACKET FOR LSI-11 SYSTEMS	
DISTRIBUTION UNRESTRICTED	
ORIGINATOR JOHN HUGHES	

Here is a product that will help in applications that require a mechanically rigid system with double-sized modules in a 4x4 backplane or any time the 9x6 backplane is used. The hold-down bar that is described in the following clipping from the 1977-78 Logic Handbook (page 369) describes the same hold-down bracket that is used on quad-sized modules, like the LSI-11 processor and the MSV11-CD memory board.

To use this bracket, customers can drill out the existing handles on double or quad-sized board and attach the bracket by means of either rivets or screws.

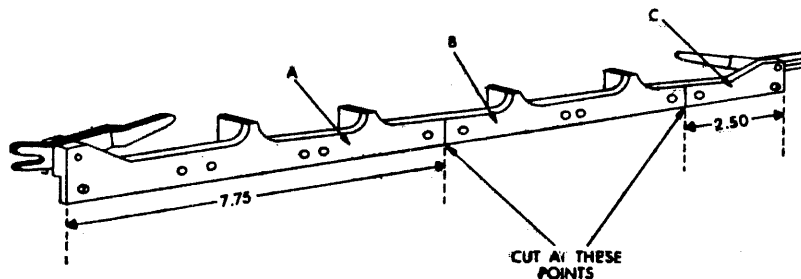
In addition to offering increased rigidity and resistance to vibration, this bracket makes insertion and removal of modules far easier.

**Hold-Down Bracket 12-10711-02**

The 12-10711-02 module hold-down bracket serves as a mechanical combination handle and hold-down bracket for Hex-size modules when used with the appropriate cards guides such as an H0341. This bracket can also be used as a hold-down bracket for LSI-11 compatible modules (quad-size) but must be modified as shown in the accompanying drawing.

**LSI-11 HOLDOWN 12-10711-02**

1. CUT BRACKET AT POINTS DESIGNATED ON DRAWING BELOW.
2. DISCARD CENTER SECTION "B".
3. MOUNT SECTIONS "A" AND "C" ON BOARD AS DESIRED.



**digital**  
**COMPONENTS**  
**GROUP**

<b>ynote</b>		NUMBER 010 A
TITLE <u>Power Supply For H909C</u>		DATE 6 /7 /77
DISTRIBUTION <u>H909C Customers</u>		PRODUCT H909C
ORIGINATOR <u>David Schanin</u>		PAGE <u>OF</u> 2

THIS MICRO NOTE REPLACES #010. #010 SHOULD BE DISCARDED AND REPLACED WITH THIS ONE AS IT WAS IN ERROR. ....

The H909C Expander Box is a convenient way to package the DDV11-B 9x6 backplane with the H0341 card guide. However, the H780 power supply may not be used with the H909C because air flow from the power supply fans will be restricted. Customers must provide their own power supplies and provide for cooling of both the LSI-11 cards and the power supplies.

There are two configurations of Lambda power supplies that have the potential of powering the H909C module systems enclosure. One is a 148-watt configuration that sells for \$400 and the other is a 321-watt configuration that sells for \$760.

The 321-watt configuration requires two +12 volt supplies to be connected in parallel. This is accomplished by connecting a diode in series with each positive output and connecting the sense input on the load side of the diode. One supply will current limit and the other will regulate. The diode is to protect the over-voltage protection circuit.

CONFIGURATION ONE

<u>QTY</u>	<u>MODEL</u>	<u>VOLTAGE</u>	<u>CURRENT</u>	<u>PRICE</u>	<u>POWER</u>
1	LJS-11-5-OV	5V	20A	\$220	100 Watts
1	LJS-10-12-OV	12V	4A	<u>\$180</u>	<u>48 Watts</u>
				\$400	148 Watts


CONFIGURATION TWO

<u>QTY</u>	<u>MODEL</u>	<u>VOLTAGE</u>	<u>CURRENT</u>	<u>PRICE</u>	<u>POWER</u>
1	LGS-5-5-OV-R	5V	45A	\$400	225 Watts
2	LJS-10-12-OV	12V	8A	<u>\$360</u>	<u>96 Watts</u>
				\$760	321 Watts

**digital**  
**COMPONENTS**  
**GROUP**

The KPV11-A can be used to generate the power fail/restore signal sequence and line time clock normally provided by the H780 power supply. An H780 control panel may also be interfaced to the LSI-11 via the KPV11-A.

**NOTE:** DUE TO PHYSICAL AND COOLING LIMITATIONS, THE H780 POWER SUPPLY CANNOT, UNDER ANY CONDITIONS, BE USED IN THE H909C BOX.

	NUMBER	011
	DATE	6 / 8 / 77
	PRODUCT	General
	PAGE1	OF2
TITLE	LSI-11 Halting During Interrupt Cycle	
DISTRIBUTION	LSI-11 Customers	
ORIGINATOR	Ted Semple	

An unusual sequence of events may cause the LSI-11 processor to HALT during an interrupt cycle. The problem occurs when an I/O device requests interrupt service simultaneously with an instruction being executed to reset the interrupt enable bit for the particular interrupt request.

An internal flip flop in the CPU chip set is set by the leading edge of an interrupt request. The processor responds by first completing execution of the current instruction and then asserting BIAK to grant the interrupt. The interrupt grant is then passed in daisy chain fashion down the bus to the board that made the request. However, if the instruction being executed resets the interrupt enable bit, the board will not realize that it was the one that generated the request and will, in turn, continue the daisy chain of the grant down the bus. If no other board has an interrupt request pending, the interrupt grant will pass all the way to the end of the bus and 12 us. later, the processor will time out and enter the HALT mode. This scenario, although unlikely, may be the cause of some previously unexplained CPU HALTs.

These false interrupts can be eliminated by modifying software. Before clearing an interrupt enable bit, all interrupts should be disabled by setting bit 07 of the PSW. To avoid needless interrupt latency, bit 07 of the PSW should be cleared as soon as possible after the interrupt enable bit is cleared.

The same scenario occurs in noisy environments when the BIRQ signal line is glitched by noise. In this case, a BIAK will be issued by the processor when no board has made an interrupt request.

There is a solution for this problem for those customers who are not using system software packages such as RT-11. This solution may be implemented by adding a single strap to the backplane and by a simple software modification.

On the last slot used on the backplane pin AN2, BIAKO, should be connected to pin AF2, BRPLY. During program loading, address location 000000 should be loaded with 000002 and location 000002 should be loaded with 000002.

These modifications will cause the following sequence to occur. When an interrupt acknowledge is not captured by a module, the interrupt acknowledge itself becomes the reply to that interrupt acknowledge. The processor, seeing a reply, will assume that a vector is on the bus. The bus, being in a floating condition, is equivalent to having a 0 vector. The processor then fetches the contents of location 000000 and stores this in the program counter, register 7 of the processor. Next, the processor fetches the contents of memory location 000002 and loads that into the processor status word. As with any interrupt cycle, the processor then begins to execute the program whose starting address is now in register 7. In this case, the processor will begin executing a program starting at memory location 000002. The instruction in this location is an RTI, Return from Interrupt. The RTI instruction causes the processor to return to the original program that was falsely interrupted, and the operation will continue as if the interrupt never occurred.

CAUTION: This solution should only be used with operating systems that do not use memory location 000000. RT-11 and other operating systems use location 000000.

<b>μnote</b>		NUMBER 012
TITLE	LSI-11 Bus Theory of Operation	DATE 7 /11 /77
DISTRIBUTION	LSI-11 Customers	PRODUCT H9270; DDV11-B
ORIGINATOR	Ted Semple	PAGE 1 OF 9

**1.0 - PROPAGATION DELAY & REFLECTIONS**

If a voltage is supplied between any two conductors, they may be considered as a transmission line. The ideal transmission line input impedance looks like pure resistance but, in fact, is mainly a combination of capacitance and inductance (see Figure 1).

When the voltage at one end of the transmission line is changed, that change does not instantaneously appear at the other end. There is some delay, which is called propagation delay (see Figure 2). The propagation delay of LSI-11 bus cable is approximately 1.4 to 1.9 ns. per foot (or .0348 m).

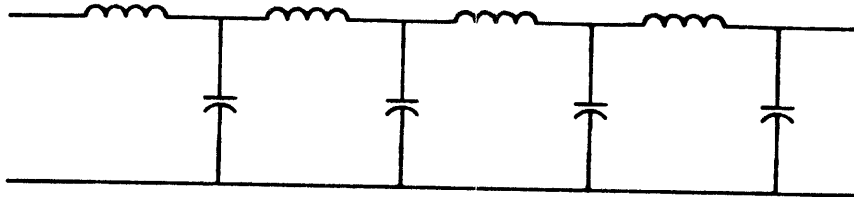


Figure 1. Transmission Line Circuit Example

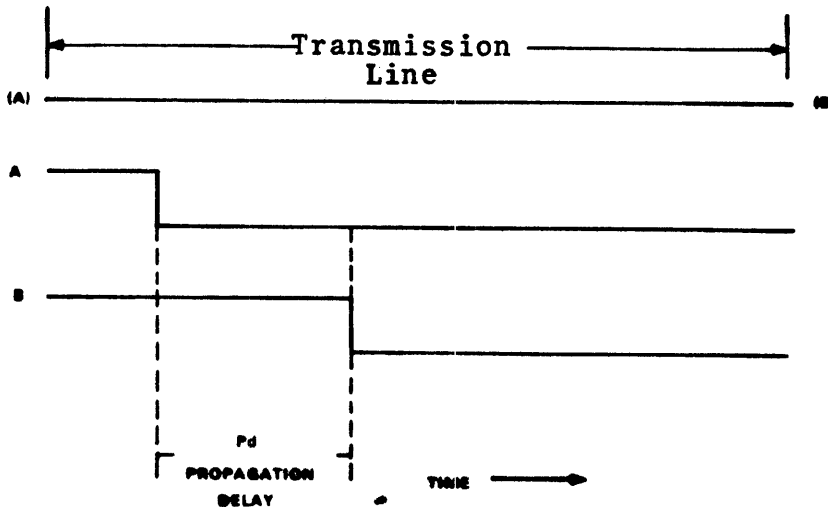


Figure 2. Cable Delay Example

A lossless transmission line of infinite length looks like a pure resistance of value  $Z = L/C$ . If such a line is broken and terminated with a resistor of this value ( $R = Z$ ), the line will behave like an infinitely long line; i.e., appear resistive (see Figure 3). The impedance ( $Z$ ) of LSI-11 bus cable is approximately 120 ohms.

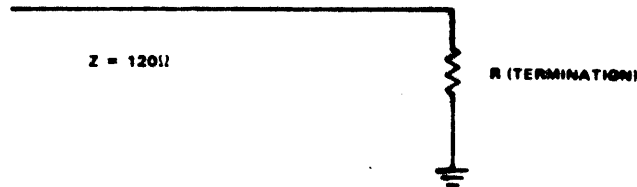


Figure 3. Cable Impedance Example

When a voltage is applied to the line, the instantaneous power consumed will be:

$$P = E^2/Z \qquad P = E^2/120$$

Suppose now that the terminating resistor ( $R$ ) is not equal to the characteristic impedance of the line. The power which is traveling down the line before it reaches the termination (during propagation delay) is equal to  $E^2/Z$ , but the power dissipated in the resistor after propagation delay is approximately equal to  $E^2/R$ .

If resistance ( $R$ ) is greater than the impedance ( $Z$ ), there will be extra energy available at the termination and since energy cannot simply disappear, it will be reflected back into the line. After one more propagation delay (for the return journey), this reflection will be seen back at the source (see Figure 4A).

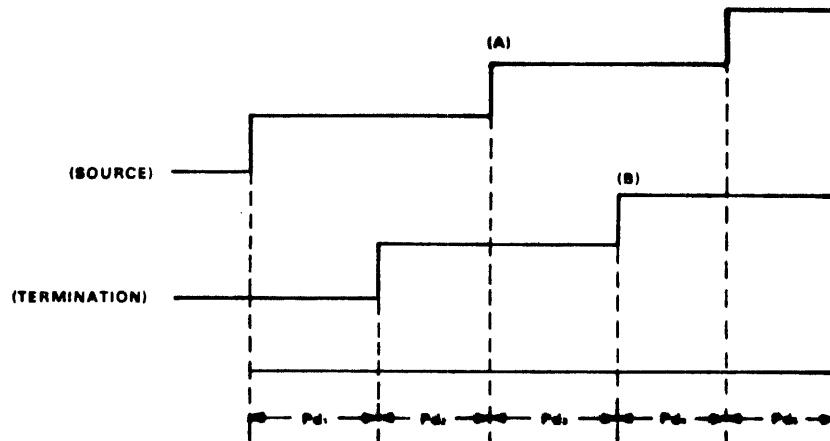


Figure 4. Impedance Mismatch Example

Some of this reflection will be dissipated by the source impedance and some will be re-reflected back into the line. When this re-reflection is seen at the termination (Figure 4B), (after still another propagation delay), the energy difference will be reflected back to the source. Eventually both source and termination will arrive at the same level. The important point to note here is that what was intended to be a level change with a clean transition did not turn out that way on the line due to a termination mismatch between R and Z.

Essentially the same situation occurs when the termination resistor is smaller than the characteristic impedance of the line. If resistance (R) is less than impedance (Z), there will be a mismatch and this will also be reflected back to the source (see Figure 5).

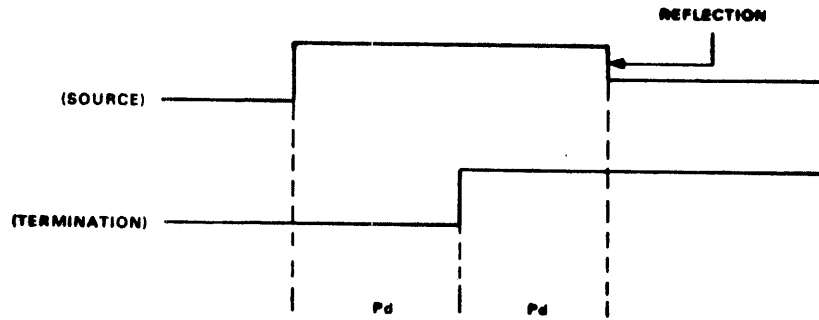


Figure 5. Impedance (Low Resistance) Mismatch Example

Note that: (a) transmission line which is not terminated in its characteristic impedance will have reflections and (b) the voltage seen at any point on the line or at any instant in time will be a combination of the incident and reflected voltage.

The amount of reflection depends on the mismatch, and approaches 100 percent for either a shorted or an open line (see Figure 6).



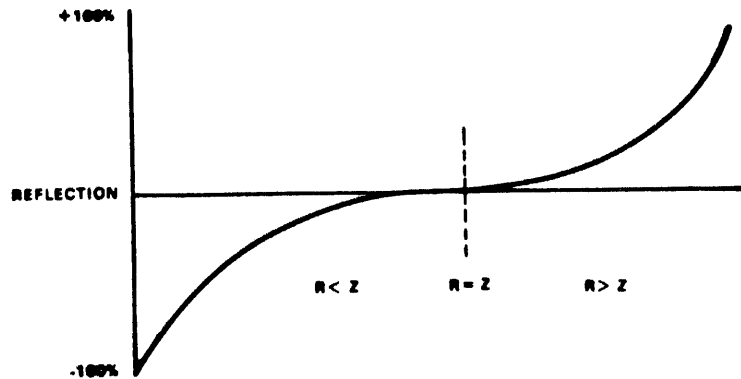


Figure 6. Mismatch Reflection Curve Example

Essentially the same thing happens on the negative going edge of a level change so that what seemed to be a clean transition as in Figure 7A may look more like Figure 7B.

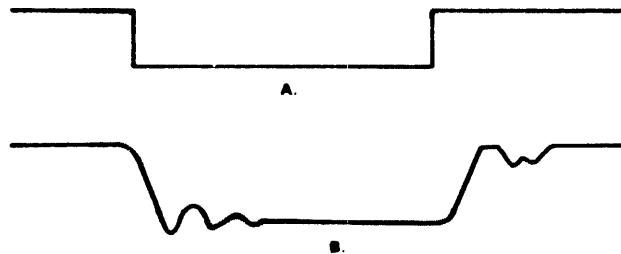


Figure 7. Cable Mismatch (Waveform Example)

To further compound the problem, each device on the bus contributes its own unique impedance and mismatch in complex time relationships (Figure 8).

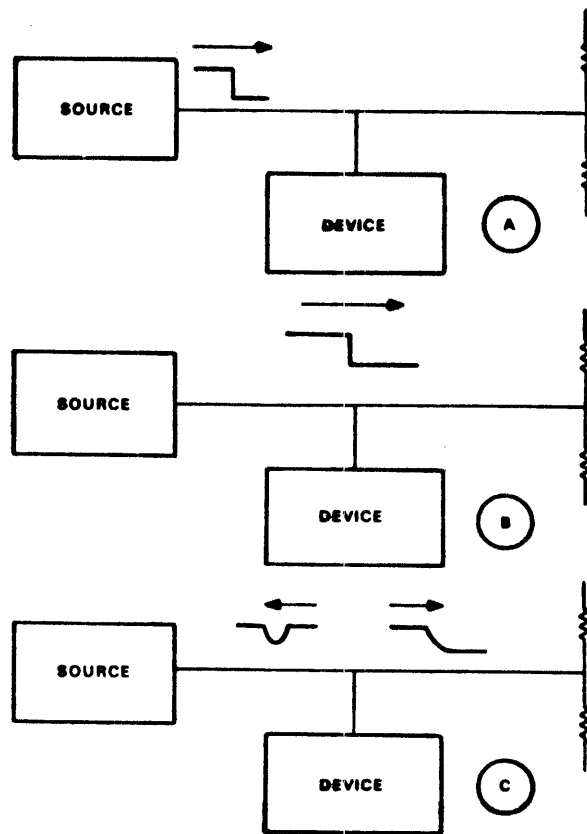


Figure 8. System Device Impedance Example

All of this impedance and resistance mismatch is normal and to be expected in LSI-11 bus systems. The objective here is to point out the possible impedance mismatch and how to use the appropriate tools to minimize the effects of reflections and "noise".

### 1.1 - LINE TERMINATION TECHNIQUE

The question may be asked, how can a 178 and 383 ohm resistor properly terminate a line which has an impedance of 120 ohms?

A perfect power supply has an AC resistance (impedance) of zero ohms, so for AC considerations the termination diagram changes somewhat to that shown in Figure 9B and simplified in Figure 9C.

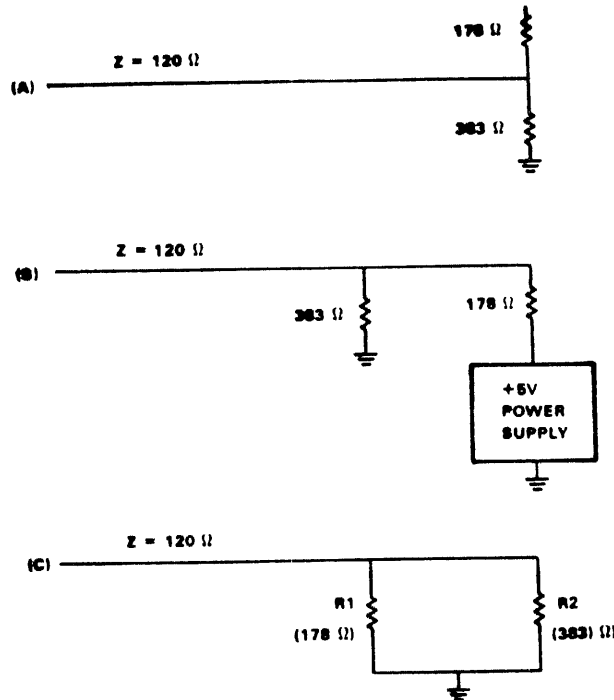


Figure 9. Line Termination Technique Example

The choice of these two values satisfies both the quiescent condition and the required termination impedance.

## 2.0 - AC LOAD

An AC load is defined as a number related to the impedance that a bus element presents to an LSI-11 bus signal line (due to backplane wiring, PC etch runs, receiver input loading, and driver output loading). This impedance load on a transmission line causes a "reflection" to occur when a step is sent down the line. This reflection shows up on an oscilloscope as a spike occurring shortly after an asserting or unasserting edge. An AC load is nominally 9.35 pf. of capacitance. Nine lumped AC loads reflect 20 percent and 20 lumped AC loads reflect 40 percent of a 25 ns. risetime step. AC loads must be distributed

on the bus in the manner described in the LSI-11 Configuration Guidelines in order to provide bus operation with reflections guaranteed to be at or less than a tolerable level. The AC load rating of LSI-11 bus elements is usually based on the greatest of the capacitances that the element presents to the BDOUT, BDIN, BRPLY, BSYNC, BREF and BSACK signal lines. If the element is customer-designed, its AC loading must be determined from a reasonable estimate of the equivalent capacitance presented to the LSI-11 bus.

3.0 - DC LOAD

A DC load is defined as a number related to the amount of DC leakage current that a bus element presents to an LSI-11 bus signal line which is high (undriven). A DC load is nominally 105 uA (80 uA - receiver plus 25 uA - driver). However, the DC load rating of a bus element is not strictly based on the element's signal line that has the greatest leakage, (e.g., DC leakage is less important on BDAL lines than it is on BSYNC). The DC loading of an element should always be obtained from the specification for that element. It should not be obtained from a calculation of the receiver and driver leakage current, unless the element is custom-designed and is not listed in the applicable documentation.

4.0 - LOADING RULES

4.1 In multiple backplane systems, each backplane can have up to 20 AC loads. If a load is too large, it may generate a reflection on the LSI-11 bus large enough to create a false logic signal and cause a failure. Figure 10 illustrates such a failure. The reflection may cause the threshold of the 8640 receiver to be crossed a second time.

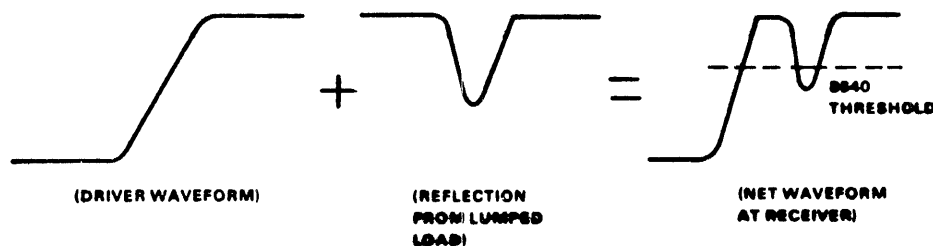


Figure 10. AC Loading Violation

To avoid this problem, some of the modules should be moved to another backplane. Best results will be obtained when the AC loading of all backplanes is equal.

- 4.2 In single backplane systems, the number of AC loads is limited by the bus termination impedance. The RC time constant of the loaded backplane determines the rise time of a signal line when a driver unasserts that line. If the maximum RC time constant is exceeded, the signal may not rise within 25 ns. (see Figure 11). 25 ns. is the maximum delay permissible and still meet bus timing requirements.

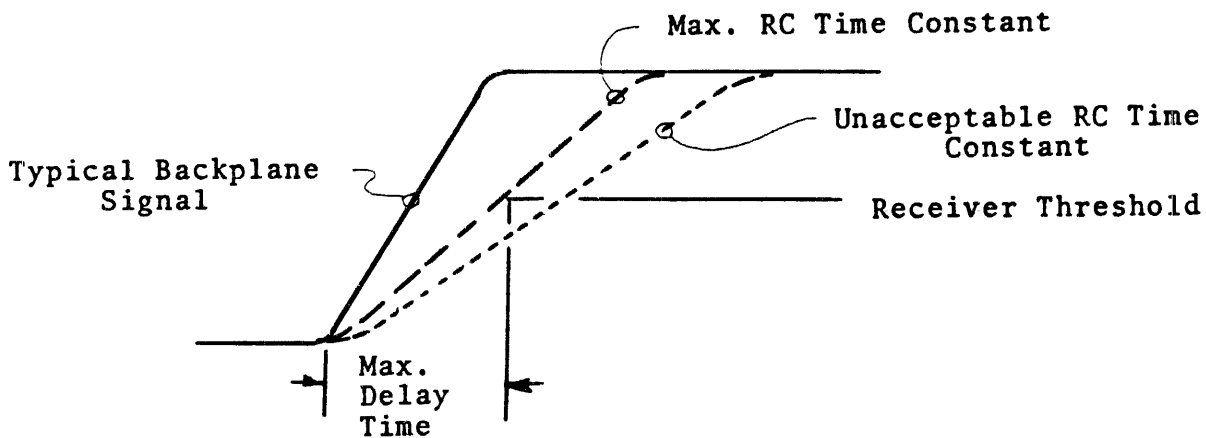


Figure 11. Single Backplane Signal Waveform

To slow a rise time may also cause the output of some bus receivers to oscillate as the input signal rises through the receivers threshold.

## 5.0 - DC LOADING RULE

The maximum number of DC loads in either a single or multiple backplane system is 20. If too many DC loads are put on the bus, the quiescent undriven voltage may be lowered to a level where bus receivers become susceptible to reflections from lumped loads and the overall noise margin on the high end (bus undriven) may become too small.

## 6.0 - CABLING RULES

In multiple backplane systems, the cables must be at least 2 feet long. If a cable is less than 2 feet long, the system will behave like a single backplane system.

In multiple backplane systems, the cables must be at least 4 feet different in length. When this rule is violated and a driver in the middle backplane unasserts the bus, reflections from the other backplanes will arrive back at the middle backplane simultaneously and superimpose. The net reflection may cross the 8640 threshold and cause a failure (see Figure 12). When the cables are different lengths, the reflections will arrive at slightly different times (see Figure 13).

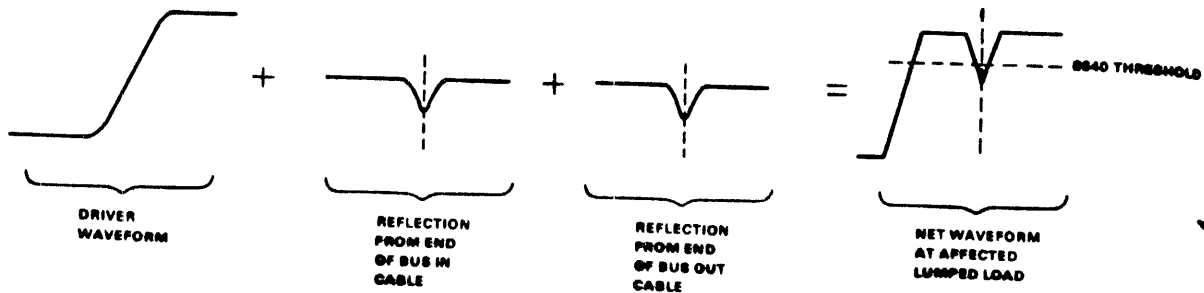


Figure 12. Violation Of Cable Length Rule Waveform

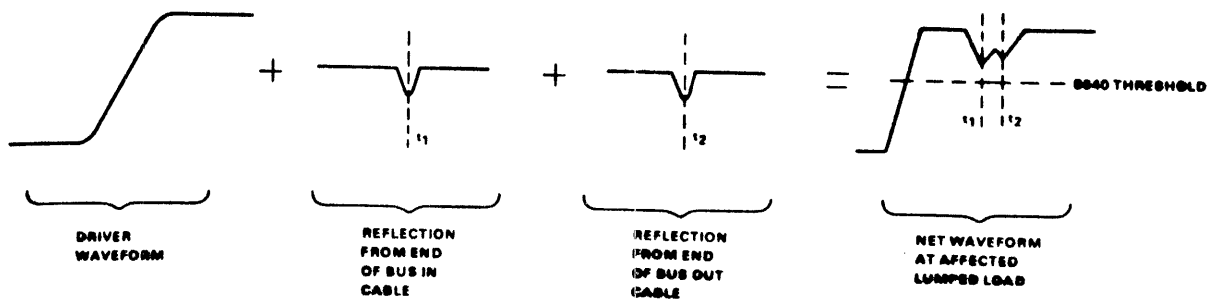



Figure 13. Proper Cable Length Rule Waveform

	NUMBER	013
	DATE	8 / 2 / 77
	PRODUCT	APL-11
	PAGE	OF 1
TITLE <u>Installing APL-11 on 11V03 &amp; 11T03</u>		
DISTRIBUTION <u>All APL-11 Customers</u>		
ORIGINATOR <u>Rich Billig</u>		


When installing RT-11 APL on an 11V03 or 11T03 system, you must consider two options:

- Choice of:
  - SINGLE PRECISION variables (7 decimal digits of precision), OR
  - DOUBLE PRECISION variables (17 decimal digits of precision), AND
- KEV11 optional hardware

Unless the application requires high precision, we recommend choosing SINGLE PRECISION for speed reasons.

To choose the correct APL interpreter file for your configuration, refer to the following table:

KEV11 PRESENT?	SINGLE PRECISION	DOUBLE PRECISION
YES	APL04.SAV	APL03.SAV
NO	APL00.SAV	APL01.SAV

 <b>Correct Input Parameters for the QJV11 PROM Formatting Program</b> <b>TITLE</b> _____ <b>DISTRIBUTION</b> <u>All QJV11 Users</u> <b>ORIGINATOR</b> <u>Dave Schanin</u>	NUMBER 014
	DATE 9 / 28 / 77
	PRODUCT QJV11
	PAGE 1 OF 2

The document that is shipped with the QJV11 PROM Formatting Program has a serious error in it. The document is in the process of being corrected, but until such time, Chapter 7 of the Microcomputer Handbook should be used as a guide to using the QJV11.

The error is in the description of the answers the user is supposed to give to questions printed by QJV11. A sample typeout is listed below:

PROM VO1-00

ENTER AN OCTAL VALUE IN RESPONSE TO QUESTIONS WHICH REQUIRE A NUMERIC RESPONSE. TYPE 'Y' FOR YES AND 'N' OR NOTHING FOR NO. TERMINATE ALL RESPONSES WITH A <CR> (CARRIAGE RETURN). RUBOUT MAY BE USED TO DELETE ONE CHARACTER AT A TIME BEFORE <CR> IS TYPED. CTRL/U MAY BE USED TO DELETE THE ENTIRE RESPONSE. CTRL/O MAY BE TYPED TO TURN OFF OUTPUT TO THE TERMINAL.

} Initial Message

HOW MANY WORDS ARE IN A PROM? \_\_\_\_\_  
 HOW MANY BITS ARE IN A PROM WORD? \_\_\_\_\_  
 HOW MANY PROMS ARE USED IN PARALLEL? \_\_\_\_\_  
 ARE THE DATA BITS INVERTED? \_\_\_\_\_  
 ARE THE ADDRESS LINES INVERTED? \_\_\_\_\_  
 HOW MANY BYTES ARE IN THE AREA TO BE OUTPUT? \_\_\_\_\_  
 WHAT IS THE STARTING ADDRESS OF THE AREA TO BE OUTPUT? \_\_\_\_\_  
 IS YOUR INPUT/OUTPUT DEVICE ON THE HIGH SPEED READER/PUNCH? \_\_\_\_\_  
 READY INPUT, TYPE <CR> WHEN READY. <CR>

} Input Parameters

DO YOU WISH TO PUNCH TAPES? Y  
 DO YOU WANT TO VERIFY A TAPE? Y  
 READY INPUT, TYPE <CR> WHEN READY. <CR>  
 DO YOU WANT A LIST OF THE PROM CONTENTS? Y  
 DO YOU WANT IT ON A LINE PRINTER? N

} QJV11 Operation


**digital**  
**COMPONENTS**  
**GROUP**



The proper responses, based on the type of PROMs used, are listed below. Note that all of this information is copied from the Microcomputer Handbook.

Table 7-5 QJV11 Input Parameters

Parameter	MRV11-AA Applications		MRV11-BA Applications
	512 × 4 PROMs	256 × 4 PROMs	1K × 8 PROMs
No. words in a PROM ( $N_p$ )	1000	400	2000
No. bits in a PROM word ( $N_b$ )	4	4	10
No. PROMs used in parallel	4	4	<u>2</u>
Are data bits inverted	N	N	N
Are addr. lines inverted	Y	Y	N
How many bytes in the area to be output ( $N_a$ )	20000	10000	20000
Starting Address	0, 20000, 40000, 60000, 100000, etc.	0, 10000, 20000, 30000, 40000, etc.	0, 20000, 40000, 60000, 100000, etc.
I/O device on the H.S. reader/punch	Y or N	Y or N	Y or N

 TITLE <u>Power Sequencing for the KD11-HA Module</u> DISTRIBUTION <u>Unrestricted</u> ORIGINATOR <u>Dave Schanin</u>	NUMBER 015
	DATE 11 / 21 / 77
	PRODUCT KD11-HA
	PAGE 1 OF 3

The KD11-HA power-up and power-down sequencing functions are exactly the same as on the KD11-F. The user may select any one of four power-up modes:

<u>Mode</u>	<u>Start-up Function</u>
0	PC at 24, PS at 26
1	ODT microcode
2	PC at 173000
3	Reserved microcode

#### Power-Up

On all KD11's, there are two methods (not modes) to start the LSI-11 and cause it to power-up through the selected mode:

Method 1: Use an H780 power supply or KPV11 power sequence module. They function as follows:

- a) DC power is applied to the backplane, while BPOK and BDCOK are asserted.
- b) 3-10 ms. following DC power application, BDCOK is released. At this time, the CPU comes up, does a fast DIN on location 4 (a DIN which ignores the actual contents of location 4), and in doing so reads the power-up jumpers and the state of BPOK. The CPU senses BPOK is asserted and continues looping.
- c) 70 ms. (min.) following the release of BDCOK, BPOK is released. The CPU, which had been looping on reading the BPOK line, now senses that BPOK has been released and jumps to the location specified by the power-up jumpers.

Method 2: Use a pushbutton. Connect a pushbutton, preferably debounced, to the BDCOK line. This pushbutton should allow BDCOK to float when the button is released, and should assert (ground) BDCOK when the button is depressed. This method functions as follows:

- a) DC power is applied. BDCOK and BPOK are released. The CPU is in an undefined state and will not run.
- b) The pushbutton is depressed following DC power application and, therefore, BDCOK is asserted. As long as BDCOK is asserted, the CPU is in a reset condition and non-functional.
- c) The button is released so BDCOK is released. The CPU comes up, does a fast DIN on location 4, reads the power-up jumpers, and the state of BPOK.
- d) Since BPOK is not asserted, the CPU jumps to the location specified by the power-up jumpers.

## Power-Down

On all KD11's, two power-down methods exist:

Method 1: Power-fail detection. This requires a KPV11 or an H780 power supply. This method allows for detection of AC loss by causing a trap through location 24. The CPU then has 4 ms. to complete its transaction and halt. This method works as follows:


- a) Upon AC loss, BPOK is asserted. This causes a trap through location 24. The power supply ride-through will maintain DC power for 4.05 ms. (min.) beyond AC power loss. The user software must complete all housekeeping functions in less than 4 ms. and halt.
- b) 4 ms. following BPOK assertion, BDCOK is asserted suspending CPU operations, locking out core memory access, and initializing peripherals.
- c) 5 us. following BDCOK assertion, DC power is lost.

Method 2: Non power-fail detection. Loss of DC power, while BDCOK and BPOK are released. Under this method, DC power is simply shut off to the backplane. The CPU may lose power anywhere in a bus or execution cycle and may cause any random event to occur on the bus or at a peripheral due to the unknown state of all devices on the bus as they

randomly lose power. This is unacceptable in a core-based system since the core contents may get "scrambled". However, in a MOS RAM system where the peripherals cannot cause damage or harm to anything should they execute a random operation, this power-down method is perfectly acceptable.

How do these power-up and power-down methods affect the new KD11-HA? For most applications, the only requirement is automatic power-up and -down, as outlined in Method 2. Full sequenced power-up and -down, Method 1, is usually only required for core based systems. Therefore, the KD11-HA incorporates a "wake-up" circuit which automatically powers up and down the CPU according to Method 2 (i.e., no power-fail detection) with no external hardware requirements. Essentially, the "wake-up" circuit consists of a single shot tied to the BDCOK line which is asserted following the application of +5 volts backplane power. Since this single shot only is tied to +5, +5 and +12 must come up within 50 ms. of each other to insure reliable power-up. See Micro Note #016 for recommended power supplies.

CAUTION: MSV11-B memories should not be used as bank 0 memory on the KD11-HA unless it is ECO REV E or higher. Failure to use at least a REV E board will result in inability of the KD11-HA to power-up (note that external refresh is required from the REV11). The reason for this is that on REV D or lower MSV11-B's, the negative voltage charge pump will not attain operating voltage before the KD11-HA does a fast DIN on location 4 to read the power-up jumpers. If location 4 does not reply, the CPU will hang, and never power-up.

 <b>μnote</b>	NUMBER 017
	DATE 11 /28 / 77
	PRODUCT KD11-HA
	PAGE 1 OF1
TITLE <u>LSI-11/2 Processor Clock</u>	
DISTRIBUTION <u>Unrestricted</u>	
ORIGINATOR <u>Ron Young</u>	

The CPU clock is a crystal controlled (non-adjustable) clock that will set the machine cycle time to 380 ns.  $\pm$  .01%. This should, at last, lay to rest any competitor's claim that we have to "tune" the clock to make the CPU work.

The tight tolerance on the clock will serve to tighten system performance between systems.

In addition, the 380 ns. figure was selected to optimize CPU--memory performance for our new memory offering without impairing performance with our older memories. This marriage of CPU and memory will serve to further enhance system performance.

NOTE: It has been brought to my attention that customers have been using "instruction loops" for timing purposes. This is not a good method for marking time. One must consider that the memory has "refresh cycles" to perform which can add time to the instruction execution time. This makes exact execution time impossible to calculate.

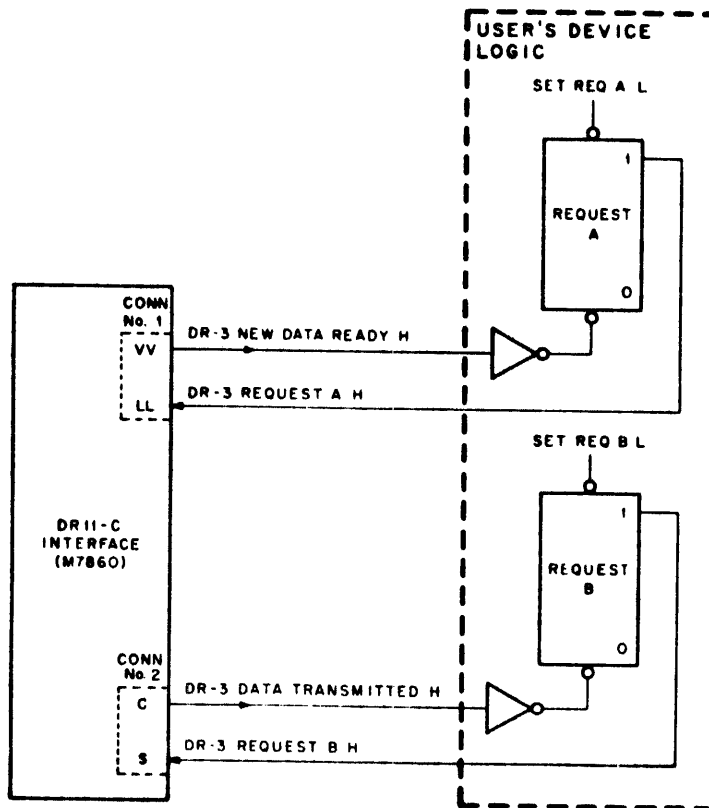
# ynote

TITLE DR11-C vs. DRV11  
DISTRIBUTION Unrestricted  
ORIGINATOR Ron Young

NUMBER	018
DATE	11 / 28 / 77
PRODUCT	DRV11
PAGE	1 OF 4

The following tables and figures list the differences between the DR11-C and the DRV11.

Some concern has been expressed on the manipulation of the REQUEST A and REQUEST B lines. Let me remind you of the "handshaking" necessary between the interface and the user's device. The user's manuals (DR11-C General Device Interface Manual--chapter 6, and ADV11-A, KVV11-A, AAV11-A, DRV11 User's Manual--chapter 5) state the need for the user to hold the REQUEST lines until the NEW DATA READY or the DATA TRANSMITTED signals are generated. The recommended method is illustrated in Figure 1 below.



**digital**  
**COMPONENTS**  
**GROUP**

TABLE I  
OUTPUT SIGNAL LOADING

<u>SIGNAL</u>	<u>DRV11</u>	<u>DR11-C</u>
New Data Ready	10 Loads **	30 Loads
Data Transmitted	30 Loads	30 Loads
Init.	10 Loads Per Connector	30 Loads Over Both Connectors
New Data Ready LO* (byte)	N/A	30 Loads
New Data Ready HI* (byte)	N/A	30 Loads
All Other Outputs	5 Loads	7 Loads

\* Byte-oriented control signals available on DR11-C only.  
ETCH REV E or later.

\*\* One load is defined as (-1.6 mA) one TTL load

TABLE II  
VARIABLE LENGTHS OF OUTPUT CONTROL SIGNALS

<u>EXTERNAL CAPACITOR</u>	<u>NEW DATA READY</u>	<u>DATA TRANSMITTED</u>
	<u>DR11-C</u>	<u>DR11-C</u>
None	350 ns.	450 ns.
470 pF.	500 ns.	600 ns.
820 pF.	600 ns.	750 ns.
	<u>DRV11</u>	<u>DRV11</u>
None	350 ns.	350 ns.
.0047 uF.	750 ns.	750 ns.
.01 uF.	1550 ns.	1550 ns.
.02 uF.	2330 ns.	2330 ns.
.03 uF.	3150 ns.	3150 ns.



TABLE III  
BERG CONNECTOR PIN DIFFERENCES

<u>SIGNAL</u>	<u>DRV11</u>	<u>DR11-C</u>
In 02	J2 H,E	J2 H
Out 02	J1 RR,NN	J2 NN
New Data Ready HI (Byte)	N/A	J1 E
New Data Ready LO (Byte)	N/A	J1 H



TITLE EIA RS-422 and RS-423  
DISTRIBUTION Unrestricted  
ORIGINATOR Ted Semple

NUMBER	019
DATE	11 / 29 / 77
PRODUCT	DLV11-J
PAGE <sub>1</sub>	OF 4

Electronic Industries Association (EIA) standards RS-422 and RS-423 have been accepted as new international standards for transmission lines between electronic equipment. These new standards offer a considerable performance improvement over traditional EIA RS-232C and current loops. Unlike RS-422 and RS-423 which were developed so that newly designed equipment could have higher performance, RS-232C and current loops grew out of existing applications and were accepted after the fact as standards. RS-232C was originally developed by the Bell System as a standard for interconnecting terminal equipment to communications equipment (modems). Current loops were the method employed to interconnect teletypewriter devices.

Figure I graphically shows the performance differences between RS-232C, RS-422 and RS-423. With RS-232C, the maximum reliable cable length is 50 feet and the maximum frequency is 20K baud. Comparing that with RS-422, you will see that RS-422 has the capability of going all the way up to 4000 feet and a maximum baud rate of 10 megabaud (although not simultaneously). Table I is a more detailed comparison of the specifications.

The new RS-422 and RS-423 specifications define the characteristics of the transmitters and receivers used to drive the transmission line as well as the characteristics of the transmission line itself. The receivers and cables used for both standards are identical, but the transmitters are different. RS-422 is a differential (balanced) line system which is capable of transmitting data at high baud rates over long distances with the high noise immunity associated with balanced line systems. RS-423 is a single-ended line system which inherently does not have the capability of a differential line.

EIA RS-423 is actually a stepping stone between EIA standards RS-232C and RS-422. RS-423 transmitters and receivers are actually backward compatible with RS-232C transmitters and receivers. This means that an RS-232C transmitter can send data to an RS-423 receiver and vice versa. Since the receivers used for RS-423 are identical to the receivers used for RS-422, it is possible to design systems which will work with existing RS-232C equipment and can then be upgraded (usually via a strapping change) to RS-422. This provides for a smooth transition between current equipment utilizing RS-232C and newly designed equipment utilizing

**digital**  
**COMPONENTS**  
**GROUP**

RS-422 without instantaneously obsoleting existing equipment. Remember that when RS-232C and RS-423 transmitters and receivers are interconnected, the system performance is that of RS-232C rather than the improved performance of complete RS-423 systems.

The DLV11-J 4-line serial line unit has been designed with the new hardware for RS-422 and RS-423. Straps have been provided which permit use of the RS-423 transmitters initially and then by restrapping, the board can be upgraded to RS-422 transmitters. The DLV11-J can, therefore, be used with RS-232C peripheral devices. In the future, new peripheral devices designed by Digital and other manufacturers will most likely utilize the new RS-422 specifications. The maximum baud rate with the DLV11-J is 38.4K baud, not the limit of either RS-422 or RS-423.

The product line has additional technical details on RS-422 and RS-423 for those who need it.

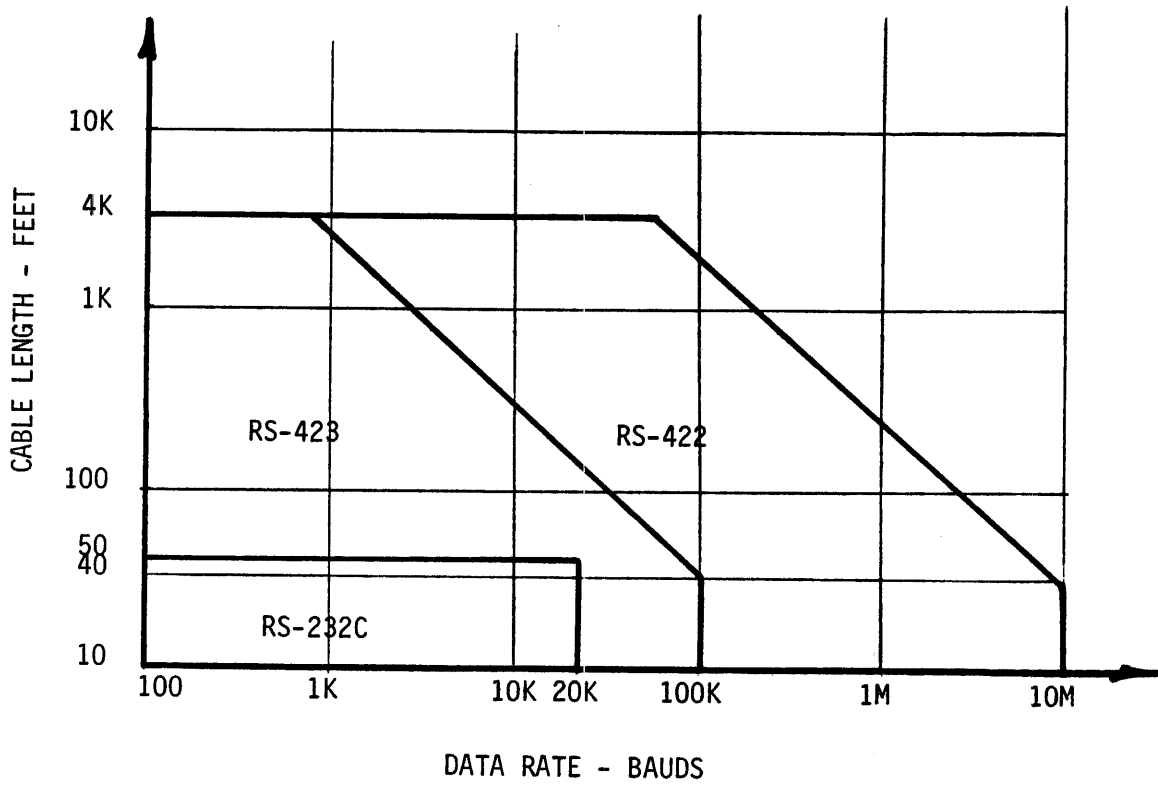



FIGURE 1 - DATA RATE vs. CABLE LENGTH

**Table 1. Comparison of the old and new interface standards**

Parameter	RS232	RS422	RS423
Line length (recommended max—may be exceeded with proper design).	50 ft	1200 m (4000 ft) See Fig. 3	1200m (4000 ft) See Fig. 5
Input Z	3 to 7 k $\Omega$ 2500 pF	> 4 k $\Omega$	> 4 k $\Omega$
Max frequency (baud)	20 kbaud	10 Mbaud	100 kbaud
Transition time* (time in undefined area between "1" and "0") tr = 10 to 90%	4% of $\tau$ or 1 ms	tr $\leq$ 0.1 $\tau$ : $\tau \geq$ 200 ns tr $\leq$ 20 ns: $\tau <$ 200 ns	tr $\leq$ .3 $\tau$ : $\tau <$ 1 ms tr $\leq$ 300 $\mu$ s: $\tau >$ 1 ms
dV/dt (wave shaping)	30 V/ $\mu$ s	See transition time	See Fig. 4
Mark (Data "1") Space (Data "0")	-3 V +3 V	A < B A > B	A = Negative B = Positive
Common mode voltage (for balanced receiver)	-	-7 V < V <sub>CM</sub> < +7 V	-
Output Z	-	< 100 $\Omega$ Balanced	< 50 $\Omega$
Open-circuit output voltage (V <sub>o</sub> ) V <sub>t</sub> = loaded V <sub>o</sub>	3 V <  V <sub>o</sub>   < 25 V  5 <  V <sub>o</sub>   < 15 V 3 to 7k $\Omega$ load	V <sub>o</sub>   $\leq$ 6 V **  2V or .5V <sub>o</sub> <  V <sub>t</sub>   † 100 $\Omega$ balanced load	4 V $\leq$  V <sub>o</sub>   $\leq$ 6 V   V <sub>t</sub>   $\geq$ .9  V <sub>o</sub>   450 $\Omega$ load
Short circuit current	500 mA	150 mA	150 mA
Power-off leakage (V <sub>o</sub> applied to unpowered device)	> 300 $\Omega$ 2 V <  V <sub>o</sub>   < 25 V V <sub>o</sub> applied	< 100 $\mu$ A 0 V < V <sub>o</sub> < 6 V V <sub>o</sub> applied	< 100 $\mu$ A  V <sub>o</sub>   < 6 V V <sub>o</sub> applied
Min receiver input for proper V <sub>o</sub>	> $\pm$ 3 V	200 mV differential	200 mV differential

\*  $\tau$  is bit period  
 \*\* across output, or output to ground  
 † whichever is greater

ELECTRONIC DESIGN 18, September 1, 1977

	NUMBER	020
	DATE	11 / 30 / 77
	PRODUCT	DDV11-B
	PAGE 1	OF 1
TITLE <u>9x6 Slot Backplane Documentation Error</u>		
DISTRIBUTION <u>DDV11-B Users</u>		
ORIGINATOR <u>Joe Austin</u>		


There is an error in the two documents which define the backplane pin assignments for the DDV11-B 9x6 slot backplane. Pins ET1 and FT1 in all slots are actually bussed to ground as shown in the table below.

These pins are not blank as indicated in the Components Group, Logic Products Option Bulletin (ED 06703 76, dated September, 1976) entitled "DDV11-B 9x6 Slot LSI-11 Backplane".

(Corrected)

**DDV11-B Backplane Pin Assignments**

SIDE	2	1	2	1	2	1	2	1
ROW	A & C	A & C	B & D	B & D	E	E	F	F
A	+5V	BSPARE 1	+5V	BDCOK H	+5V	BLANK	+5V	BLANK
B	-12V	BSPARE 2	-12V	BPOK H	-12V	BLANK	-12V	BLANK
C	GND	BAD 16	GND	SSPARE 4	GND	BLANK	GND	BLANK
D	+12V	BAD 17	+12V	SSPARE 5	BLANK	BLANK	BLANK	BLANK
E	BDOUT L	SSPARE 1	BDAL 2 L	SSPARE 6	BLANK	BLANK	BLANK	BLANK
F	BRPLY L	SSPARE 2	BDAL 3 L	SSPARE 7	BLANK	BLANK	BLANK	BLANK
H	BDIN L	SSPARE 3	BDAL 4 L	SSPARE 8	BLANK	BLANK	BLANK	BLANK
J	BSYNCL	GND	BDAL 5 L	GND	BLANK	BLANK	BLANK	BLANK
K	BWTBT L	MSPARE A	BDAL 6 L	MSPARE B	BLANK	BLANK	BLANK	BLANK
L	BIRQL	MSPARE A	BDAL 7 L	MSPARE B	BLANK	BLANK	BLANK	BLANK
M	BIAK I L	GND	BDAL 8 L	GND	BLANK	BLANK	BLANK	BLANK
N	BIAK O L	BDMR L	BDAL 9 L	BSACK L	BLANK	BLANK	BLANK	BLANK
P	BBS7 L	BHALT L	BDAL 10 L	BSPARE 6	BLANK	BLANK	BLANK	BLANK
R	BDMG I L	BREF L	BDAL 11 L	BEVNT L	BLANK	BLANK	BLANK	BLANK
S	BDMG O L	PSPARE 3	BDAL 12 L	PSPARE 4	BLANK	BLANK	BLANK	BLANK
T	BINIT L	GND	BDAL 13 L	GND	BLANK	gnd	BLANK	gnd
U	BDAL O L	PSPARE 1	BDAL 14 L	PSPARE 2	BLANK	BLANK	BLANK	BLANK
V	BDAL 1 L	+5B	BDAL 15 L	+5B	BLANK	BLANK	BLANK	BLANK

 <b>μnote</b>	NUMBER	021
	DATE	12 / 05 / 77
	PRODUCT	-----
	PAGE	1 OF 5
TITLE	Comparison of Data Transmission Techniques	
DISTRIBUTION	Unrestricted	
ORIGINATOR	Ted Semple	

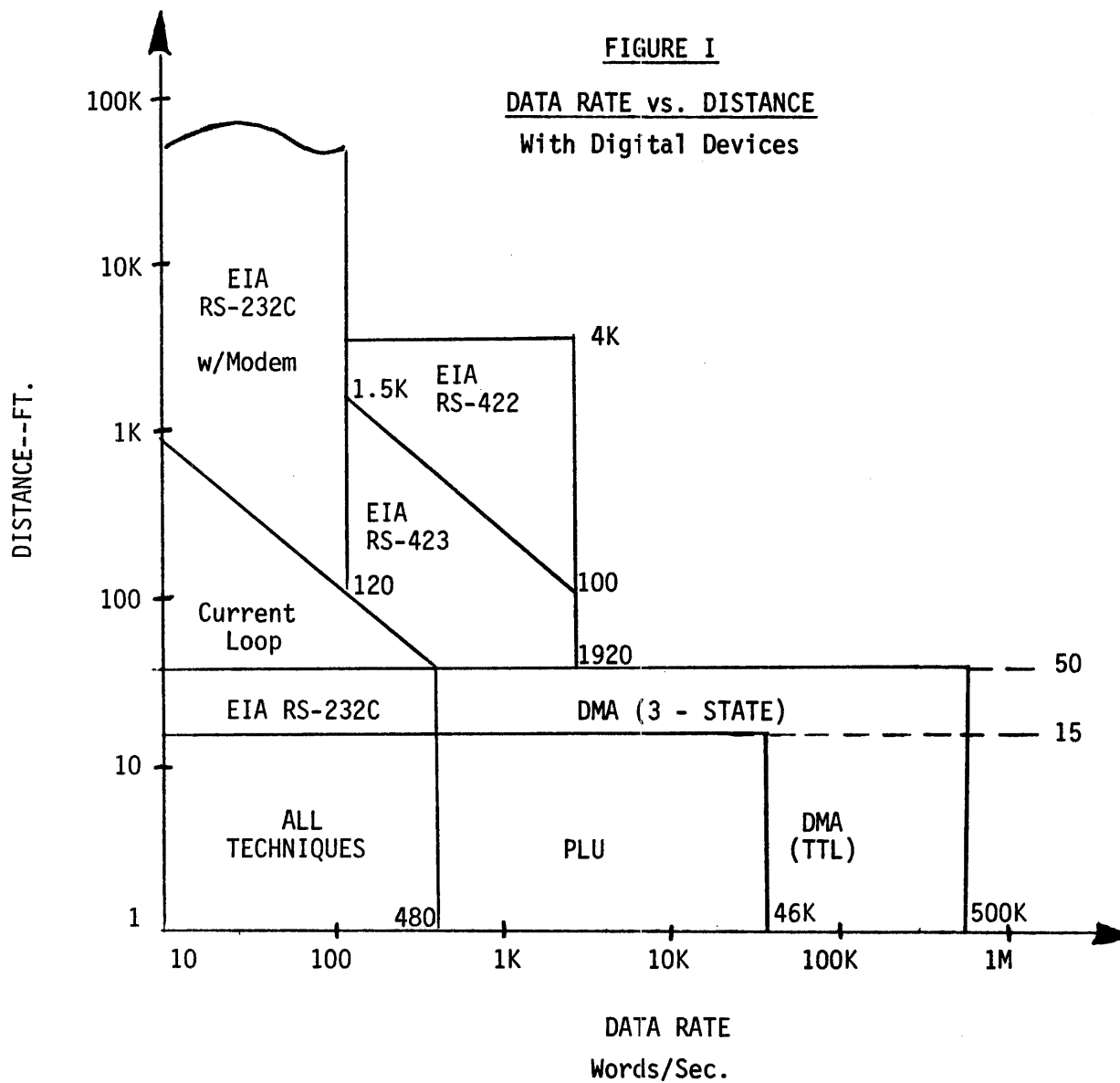
Frequently, the application arises where a data transmission path has to be established between two devices. Usually the distance between the devices is known, and also the rate of data transmission is known. The problem comes with deciding which is the best communication technique to use to interconnect the devices. Figure I should help you with this decision.

Figure I is a graph of data rate vs. distance for the various standard transmission techniques. Parallel data transmission techniques (PLU's and DMA) give the highest data rate; however, they are only good for relatively short distances. The serial techniques (RS-232C, RS-422 and current loops) are good for longer distances but at limited data rates.

While analyzing Figure I, remember that the axes are logarithmic and that the data rate is in words per second rather than baud rate. The limits established for both distance and data rate are a function of both the inherent limitations of the transmission technique and of the Digital Equipment Corporation device used to do the interconnection. As an example, look at the 422 section of the graph. Maximum distance is 4000 feet as established by the EIA standard RS-422, but the maximum data rate of 1920 words per second is based on the maximum baud rate of the DLV11-J which is 38.4K baud.

Table I is a summary of the LSI-11 devices which can be used with each communication technique. The Unibus equivalent for each device is also shown. Currently, there is no Unibus device for EIA RS-422.

The material for this Micro Note was extracted from the "CPU Interconnection Techniques" session given at the DCG International Sales Meeting. The entire presentation is available in 35 mm. slides, together with backup material, from the LSI-11 Presentation Library. Contact the product line if you would like further information.





NOTES AND ASSUMPTIONS FOR FIGURE 1


1. Data Rate Definition
  - a. One word equals 16 bits
  - b. For serial techniques, one word equals two characters formatted with one start bit, eight data bits and one stop bit. Asynchronous serial transmission is assumed.
  
2. Serial Line Maximum Data Rate
  - a. Modems were limited to 120 words/sec. (2400 baud) because modems with higher rates cost more than LSI-11 systems usually warrant. Higher data rate modems are generally synchronous rather than asynchronous.
  - b. 480 words/sec. is equal to 9600 baud, the limit of the DLV11 SLU.
  - c. 1920 words/sec. is equal to 38.4K baud, the limit of the DLV11-J SLU.
  
3. PLU (Parallel Line Unit) Limits
  - a. The TTL inputs/outputs of the DRV11 limit the distance to 15 feet.
  - b. 46K words/sec. assumes non-interrupt driven program servicing with bit testing (TSTB, BMI, MOV and SOB). 97K words/sec. is maximum rate with program servicing without bit testing (MOV and BR). With interrupt driven servicing, the maximum limit is 20K words/sec. assuming 50 us. for interrupt latency and software servicing of interrupt. (380 ns. CPU microcycle time)
  
4. DMA (Direct Memory Access) Limits
  - a. The DRV11-B can be used up to 50 feet because it has tri-state drivers and receivers. The distance is limited to 15 feet with TTL devices like the DR11-B.

- b. DMA transfers with the DRV11-B and the DR11-B are limited to 500K words/sec. in burst mode operation. 250K words/sec. is the limit for single cycle mode operation with either device. These limits are device dependent; they are not LSI-11 bus limits (which is 833K words/sec.). Remember that burst mode can disrupt memory refreshing if bus refreshing (DMA or microcode) is used. Self-refreshing memories, MSV11-CD or MSV11-D, eliminate this problem.

TABLE I

DEVICES

	<u>LSI-11</u>	<u>PDP-11</u>
Loop	DLV11	DL11-C
EIA (RS-232C)	DLV11	DL11-D
EIA W/Modem	DLV11-E	DL11-E
RS-422	DLV11-J	---
PLU	DRV11	DR11-C
DMA	DRV11-B	DR11-B

 TITLE <u>Using the MSV11-D 30K Option</u> DISTRIBUTION <u>Unrestricted</u> ORIGINATOR <u>Rich Billig</u>	NUMBER 023
	DATE 12 /16 /77
	PRODUCT MSV11-D
	PAGE 1 OF 1

The MSV11-D memory, when configured with the full complement of 16K RAM chips, provides 32K words of storage. In the LSI-11 memory map, however, the top 4K words (addresses 160000 to 177777) are normally reserved for I/O device registers (the so-called "I/O Page").

The 32K MSV11-D, as delivered, operates as a 28K word memory for the LSI-11. In this configuration, it is totally compatible with all DEC-supplied software. The 4K word region which is located at address 160000 is present but inaccessible to the program.

To allow more of the memory to be used (for large applications), a jumper-selectable option on the MSV11-D makes 30K words (rather than 28K) addressable by the program. This is done by "removing" the low 2K word area of the I/O page (addresses 160000 to 167777). When the 30K option is enabled, the memory map is:

```


000000 to 167777   -   RAM Memory
170000 to 177777   -   I/O Page

```

Because current system software expects a 4K word I/O page, RT-11 and RSX-11S cannot make use of the extra 2K words of RAM. (A user-written program operating under either system may, however, access locations 160000 to 167777 directly if desired.)

Engineering is presently evaluating the feasibility of supporting the full 30K RAM in the next release of RT-11 (V3B). Similar modifications to RSX-11S would be more extensive and are not currently planned.

When the 30K option is used, all I/O devices must be configured to place their I/O page addresses in the range 170000 to 177777. (This would not be the default assignment for such devices as the DUV11 and DZV11.) Also, it is not possible to use the REV11 bootstrap with this option, as a portion of the bootstrap PROM code resides at addresses 165000 to 165777. The new quad bootstrap module (BDV11) may be used in the 30K environment.

		NUMBER 024A
		SUPERSEDES $\mu$ NOTE #024
TITLE <u>Async., Serial Line Unit Comparisons</u>		DATE 10 / 13 / 78
DISTRIBUTION <u>LSI-11 Users</u>		PRODUCT Async. SLU's
ORIGINATOR <u>Joe Austin</u>		PAGE 1 OF 5

Attached are charts comparing the different members of the families of asynchronous serial line products. All modules of the DLV11 series are dual height modules. The DLV11-E, -F, and -J modules detect overrun conditions which are reported in the receiver CSR. These modules will not generate phantom interrupts on overrun.

#### DLV11-J

Each of the 4 serial ports on this module are separate and independent from the others. This is not a multiplexed module. Each port has its own CSR's, data buffers, interrupt vectors, baud rates, UARTs, etc. The net effect of this module is to achieve a 4:1 compression ratio over the DLV11-F and the DLV11. The main functional difference between the ports of the DLV11-J and the DLV11 is that the DLV11-J provides the higher performance EIA RS-422 and RS-423 interfaces (RS-423 on the DLV11-J also meets the EIA RS-232C specification) and requires the DLV11-KA option (one per port) to add the 20 mA current loop interface, to add 110 baud, and to add reader run functions.

#### DLV11-E

This module is functionally equivalent to the DL11-E except that it has programmable transmit baud rates. This module provides one serial port that has full modem control.

#### DLV11-F

This module is functionally equivalent to the DLV11 and will eventually replace it. In addition, it can be configured to provide programmable transmit baud rates.

#### DLV11-KA

This option allows an EIA port to be connected to a terminal with a 20 mA interface, such as an ASR33. It consists of a small PC card containing the interface conversion logic within a box that is externally mounted. Mating cables are included. This module is not restricted to 110 baud.

**TABLE 1**  
**COMPARISON OF HARDWARE FEATURES**

	UNIBUS						LSI-11 BUS						
	DZ11-A	DL11-A	DL11-B	DL11-C	DL11-D	DL11-E	DL11-W	DLV11	DLV11-E	DLV11-F	DLV11-J	DLV11-KA	DZV11-A,B
Number of Ports Per Module	8	1	1	1	1	1	1	1	1	1	4	1	4
EIA RS-232C													
Full Modem Control	✓					✓			✓				✓
Limited Modem Interface			✓		✓			✓		✓	✓		
EIA RS-422, RS-423													
Data Leads Only											✓		
20 mA Current Loop		✓		✓			✓	✓		✓		✓	
RCVR Active or Passive								✓		✓	(2)	✓	
XMIT Active or Passive								✓		✓	(2)	✓	
XMIT Active Only		✓			✓								
CCITT	✓		✓		✓	✓							✓
HALT on Framing Error (4)								✓		✓	(6)		
BOOT on Framing Error (4)									✓	✓	(6)		
Baud Rates (see Table 3)													
Programmable													✓
Split Speed Clocks Included		✓	✓	✓	✓	✓			✓	✓			✓
Reader Run Control		✓		✓			✓	✓	✓	✓	(2)	✓	
Error Flags	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
Transmit Break Generation Bit	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓
Receiver Active Bit		✓	✓	✓	✓	✓	✓		✓	✓	✓		✓
Maintenance Bit	✓	✓	✓	✓	✓	✓	✓	(3)	✓	✓	(3)		✓
Internal Real Time Clock													✓
UART Cleared by INIT	✓	✓	✓	✓	✓	✓			✓	✓	✓		✓
UART Cleared by DCOK								✓					✓
No Trap on Write to Input Buffer		✓	✓	✓	✓	✓	✓		✓	✓	✓		✓
Easy Configuration Using Wire-Wrap Jumpers									✓	✓	✓		
Stop Bits													
1	✓			✓	✓	✓		✓	✓	✓	✓		✓
1.5	✓			✓	✓	✓		✓					✓
2	✓	✓	✓	✓	✓	✓		✓	(5)	(5)	✓		✓

TABLE 2  
COMPARISON OF SOFTWARE FEATURES

REGISTER	BIT	NAME	UNIBUS						LSI-11 BUS				
			DL11-A	DL11-B	DL11-C	DL11-D	DL11-E	DL11-W	DLV11	DLV11-E	DLV11-F	DLV11-J	
RCSR:	15	data set status/interrupt					✓		(1)	✓			
	14	ring					✓			✓			
	13	CTS					✓			✓			
	12	CD					✓			✓			
	11	receiver active	✓	✓	✓	✓	✓	✓		✓	✓		
	10	2d receive	✓	✓	✓	✓	✓	✓		✓	✓		
	9,8,4	unused	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
	7	receive done	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	6	receive int. enbl.	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	5	data set int. enbl.					✓			✓			
	3	2d XMT					✓			✓			
	2	RTS					✓			✓			
	1	DTR					✓			✓			
0	rdr. enable	✓		✓			✓		✓		✓	(2)	
RBUF:	15	error			✓	✓	✓	✓		✓	✓	✓	✓
	14	OE			✓	✓	✓	✓		✓	✓	✓	✓
	13	FE			✓	✓	✓	✓		✓	✓	✓	✓
	12	PE			✓	✓	✓	✓		✓	✓	✓	✓
	8-11	unused	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	0-7	receive data	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
XCSR:	8-15	unused	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	7	XMT ready	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	6	XMT int. enbl.	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	1,3,4,5	unused	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	2	maintenance	✓	✓	✓	✓	✓	✓	(3)	✓	✓	✓	(3)
	0	XMT break			✓	✓	✓	✓	✓	✓	✓	✓	✓
XBUF:	8-15	unused	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	0-7	XMT BUF	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

**TABLE 3**  
**BAUD RATES**

BAUD RATE	UNIBUS						LSI-11 BUS						
	DZ11-A	DL11-A	DL11-B	DL11-C	DL11-D	DL11-E	DL11-W	DLV11	DLV11-E	DLV11-F	DLV11-J	DLV11-KA	DZV11-A, B
EXT								✓		✓	✓	✓	
50	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓
75	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓
110	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	(2)	✓	✓
134.5	✓			✓	✓	✓		✓	✓	✓			✓
150	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
200				✓	✓	✓		✓					
300	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
600	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
1200	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
1800	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓
2000	✓								✓	✓			✓
2400	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
3600	✓								✓	✓			✓
4800	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
7200	✓			✓	✓	✓			✓	✓			✓
9600	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
19200									✓	✓	✓		
38400											✓		



NOTES

1. Read only bit will not generate an interrupt.
2. The external 20 mA option (DLV11-KA) is required to implement this function.
3. The loop-back cable is required to implement this function.
4. Optional feature.
5. 110 baud only.
6. Applies only to the part assigned to the console device.



requires RAM from 0 to 24K and then PROM from 24K to 28K, he would have to configure the system using one 16K module, one 8K module, and one 4K PROM module.

Figure 1 shows the standard LSI-11 memory map and a sample system with 24K of RAM and 4K of ROM configured with a single MSV11-D (32K version) and a 4K PROM board.

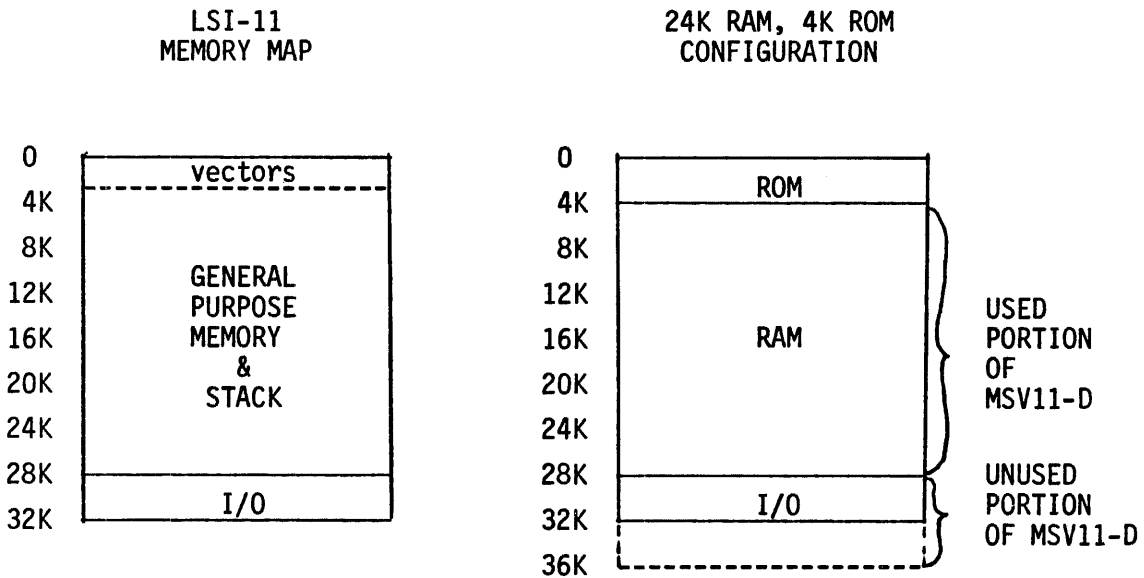



FIGURE 1

 <b>TITLE</b> <u>Micro Backplane Mechanical Mounting Guidelines</u> <b>DISTRIBUTION</b> <u>LSI-11/2 Backplane Users</u> <b>ORIGINATOR</b> <u>Chi Lau</u>	<b>NUMBER</b> 026
	<b>DATE</b> 1 / 4 / 8
	<b>PRODUCT</b> H9281
	<b>PAGE1</b> OF8

### MICRO BACKPLANES

#### FEATURES

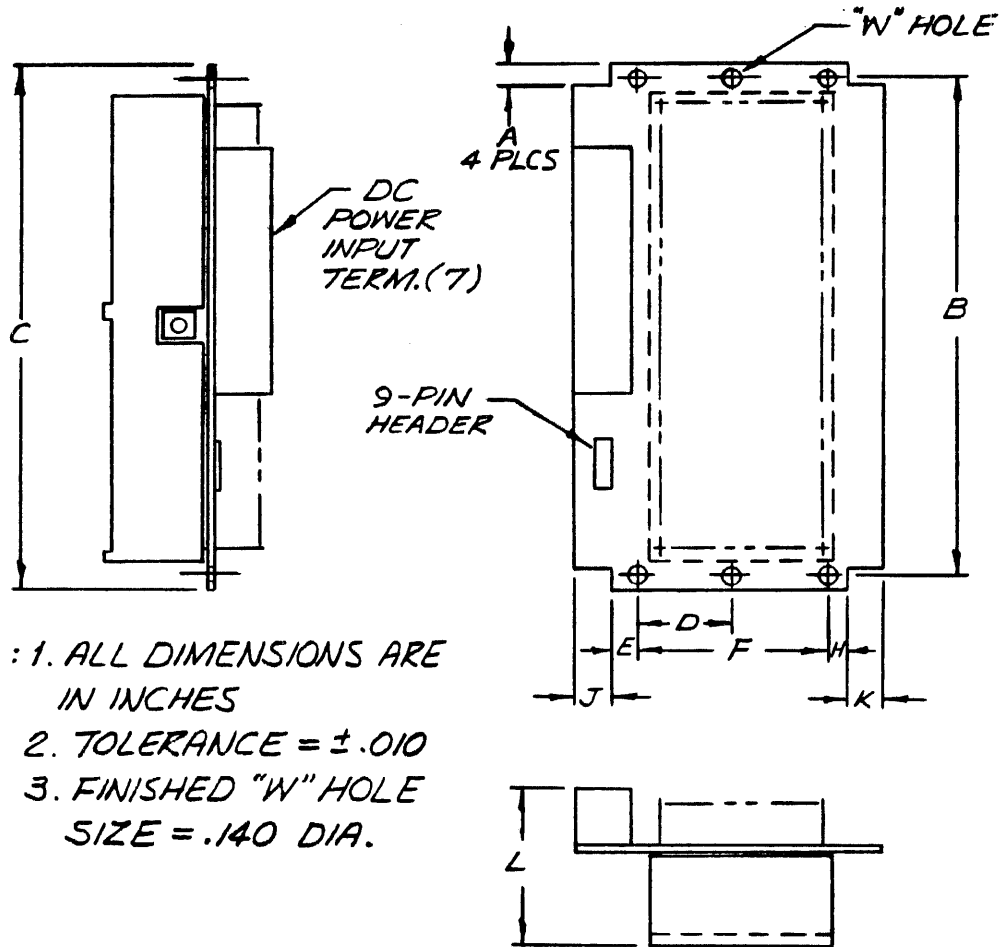
The micro backplanes are a series of backplanes with available card frame assemblies which have the following primary characteristics:

- LSI-11 bus signal connections
- Accommodates double height DEC standard modules
- Variations for four, eight, or twelve modules
- Seven terminals for DC input power (H780 compatible) including battery backup
- 9-pin header connects power supply signals to LSI-11 bus (H780 compatible)

#### DIMENSIONS AND MOUNTING

Mechanical dimensions and mounting can be divided into two categories. One is the backplane series; the other is the card frame assembly series.

1. Backplane series (H9281-AA, H9281-AB, H9281-AC) mechanical information for mounting is shown in Figure 1 and Table 1.



NOTES : 1. ALL DIMENSIONS ARE  
IN INCHES  
2. TOLERANCE =  $\pm .010$   
3. FINISHED "W" HOLE  
SIZE = .140 DIA.

FIGURE 1

TABLE 1

Option	DIMENSION										# HOLES
	A	B	C	D	E	F	H	J	K	L	
H9281-AA	.220	5.31	5.75	N/A	.180	1.74	.180	.440	.220	1.94	4
H9281-AB	.220	5.31	5.75	1.90	.280	3.80	.280	.820	.820	1.94	6
H9281-AC	.220	5.31	5.75	2.90	.280	5.80	.280	.820	.820	1.94	6

## 1.1 Stand-Off Mounting

These backplanes may be mounted onto any plane by using the mounting holes ("W" holes) in the printed circuit board. Figure 2a exemplifies this mounting scheme.

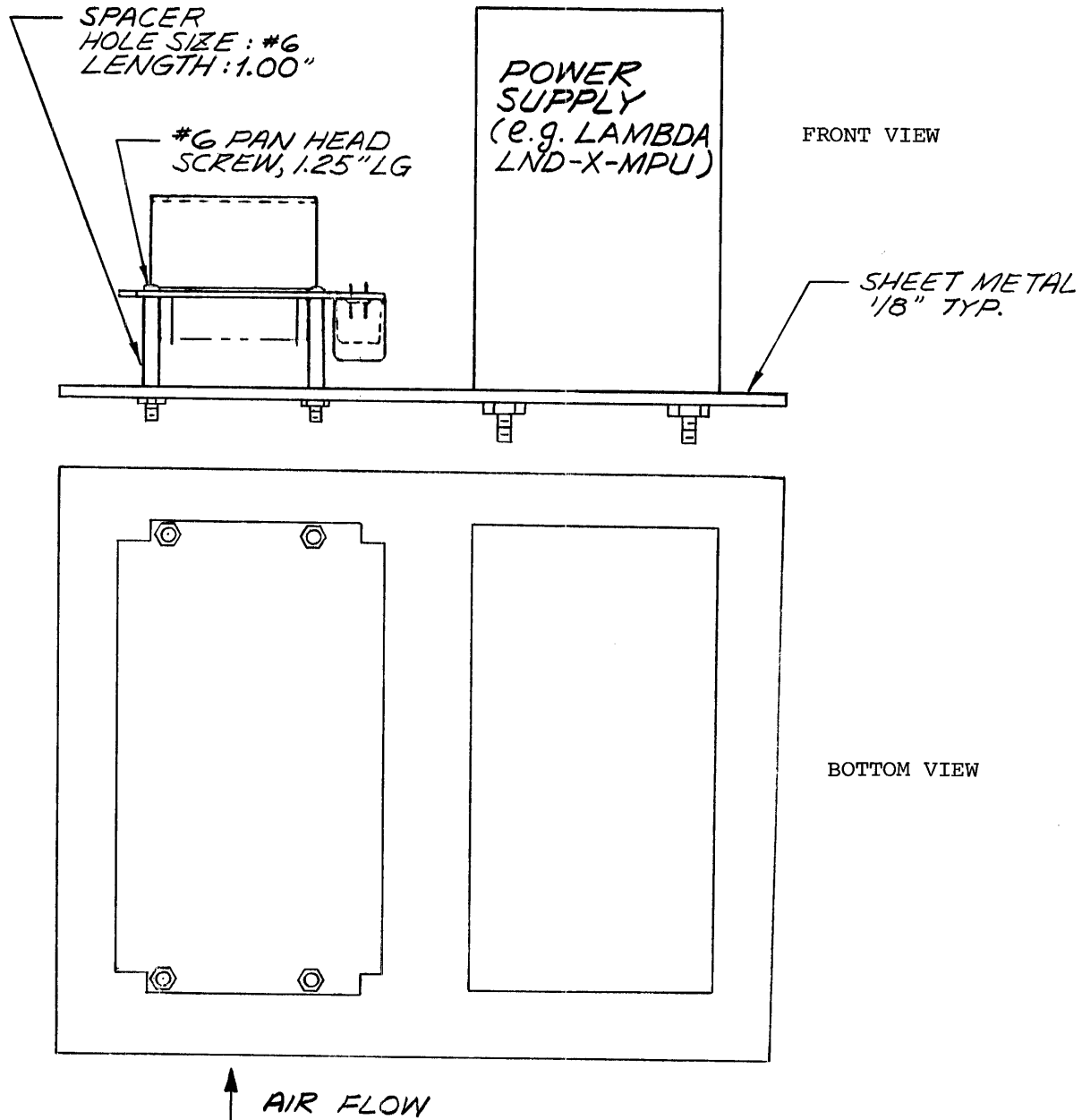
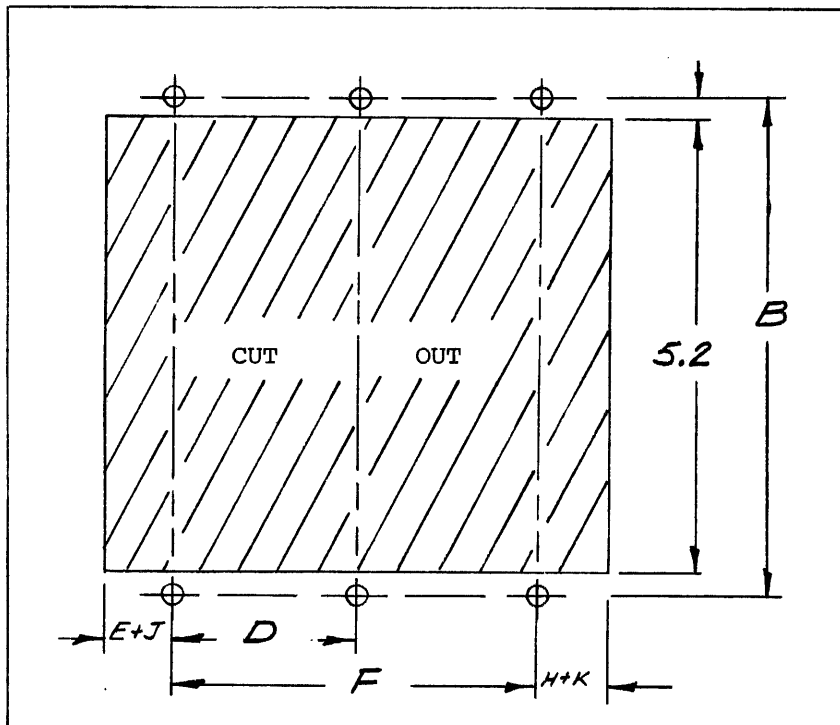
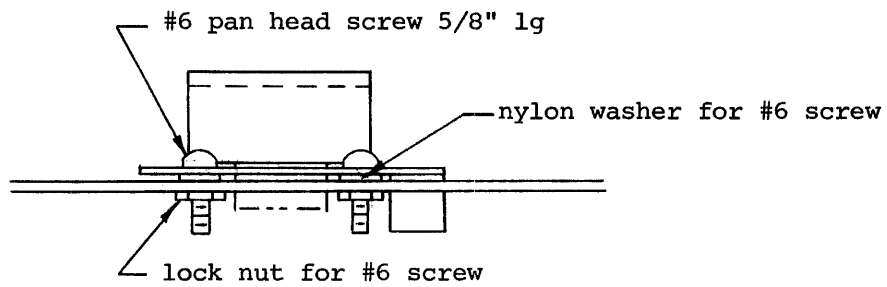


FIG. 2a ILLUSTRATIVE BACKPLANE MOUNTING TO HORIZONTAL PLANE.

## 1.2 Flush Mounting

Figure 2b exemplifies this mounting scheme. See Figure 1 and Table 1 for dimensional information.



Surface with cut-out area shaded.  
See Table 1 for dimensions.

Figure 2b - Flush Mounting

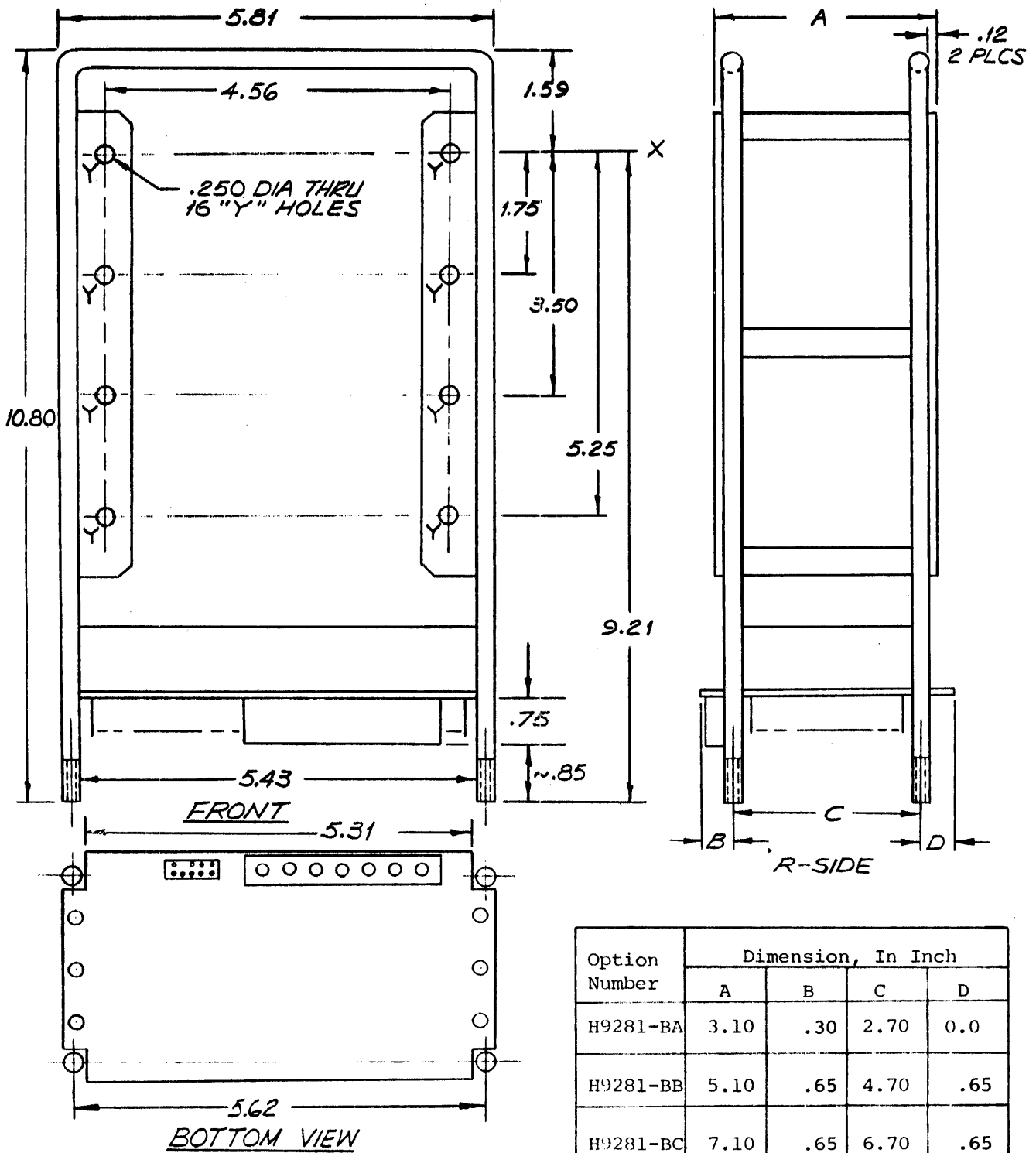


Figure 3 - Card-Frame-Assembly Mechanical Drawings (Not To Scale)



2. Card frame assembly series (H9281-BA, H9281-BB, H9281-BC) mechanical information is shown in Figure 3. These assemblies may be mounted onto any surface or support by utilizing either the threaded extensions or the "Y" holes of the card frame.

2.1 Rear-Mounting (Figure 4a)

This mounting scheme makes use of the threaded extensions of card frame.

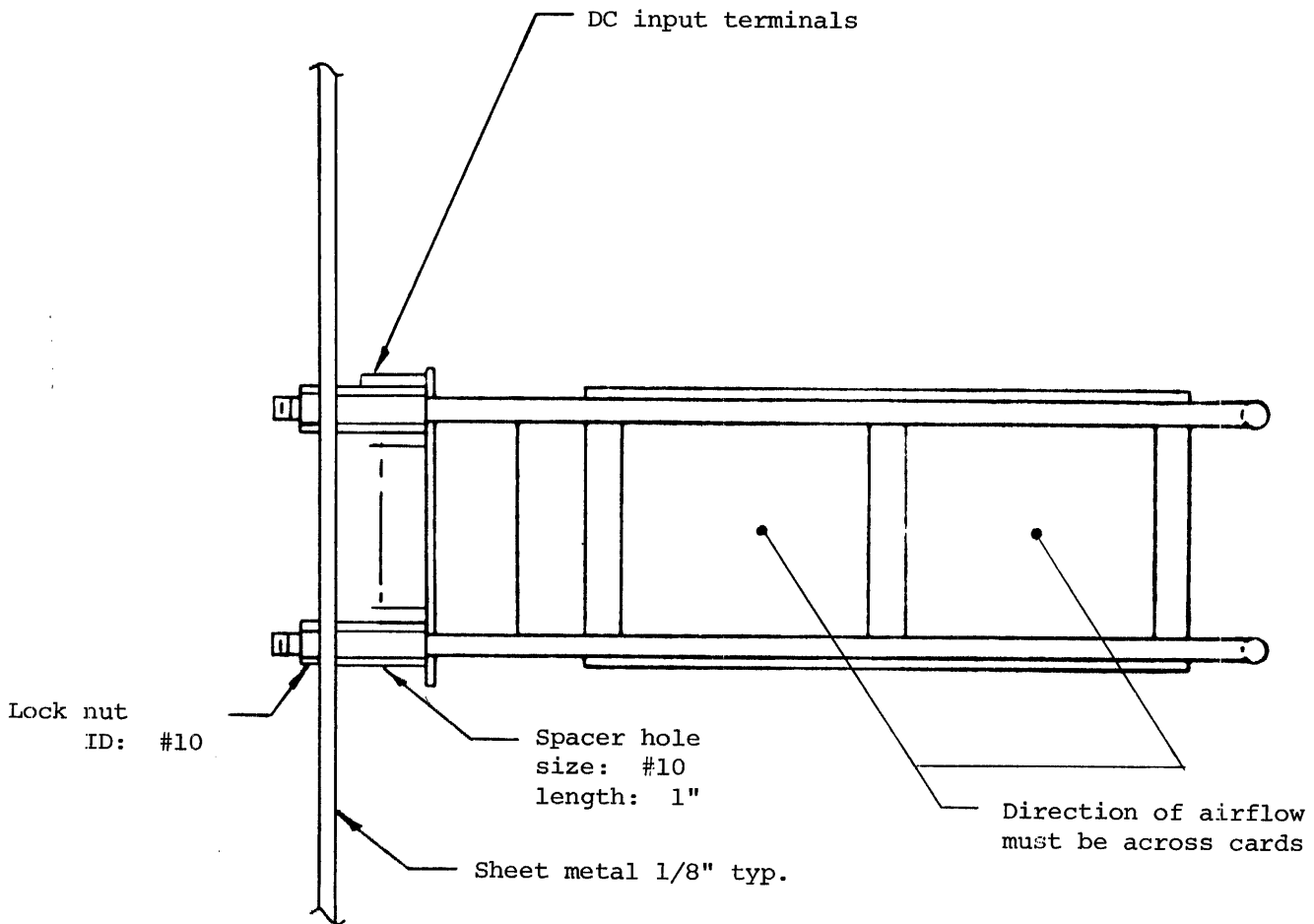


Figure 4a - Illustrative Card Frame Rear-Mounting

## 2.2 Side-Mounting

This mounting scheme utilizes the "Y" holes (see Figure 3) to mount the card frame onto a surface or side wall of an enclosure. Direction of air flow must be parallel to and through the spaces among modules.

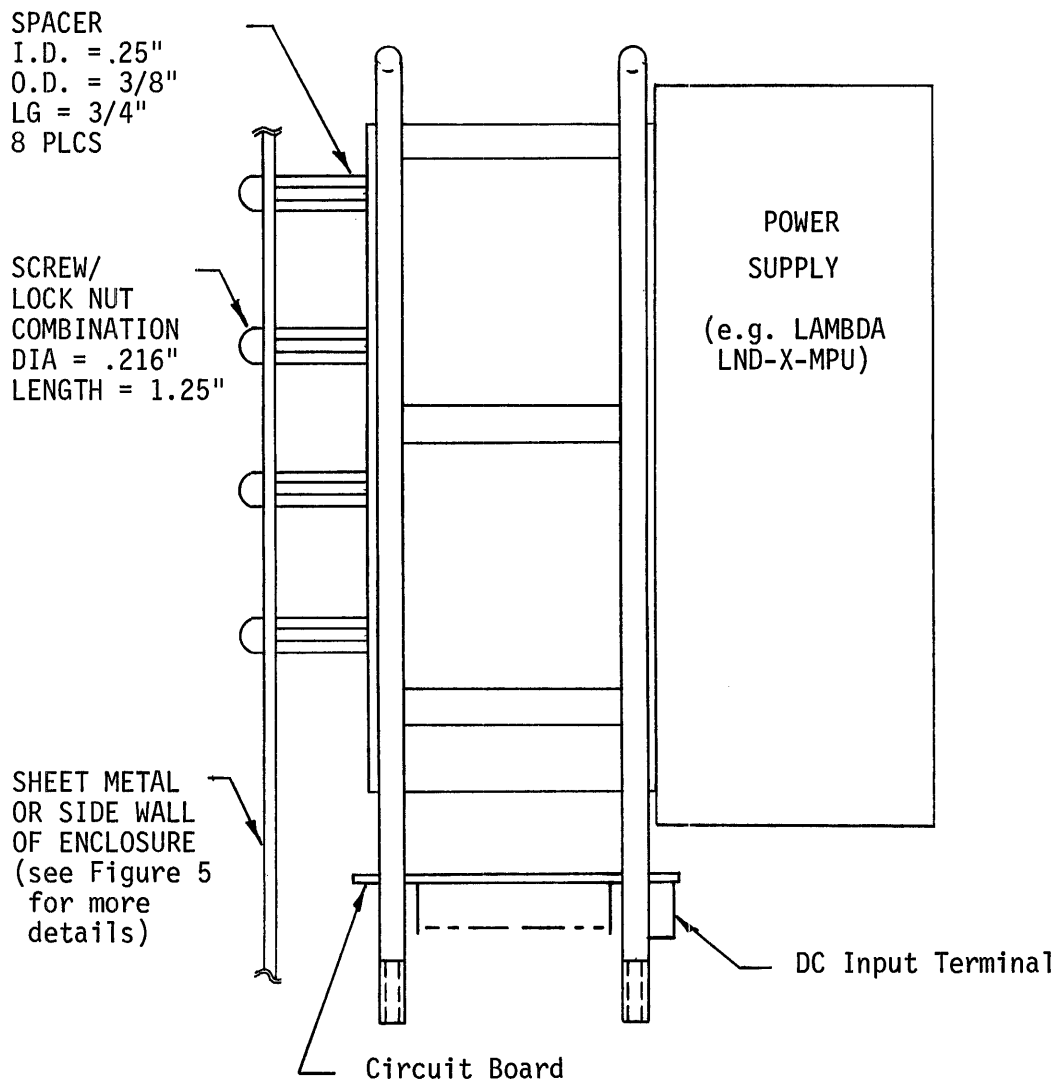


Figure 4b - Illustrative Card Frame Side-Mounting

The hole pattern for side-mounting is shown in Figure 5. Side-mounting may be accomplished in two ways. One requires spaces (see Figure 4b) to offset circuit board overlap. The other is flush mounting which requires cut-out in the mounting surface (see shaded area).

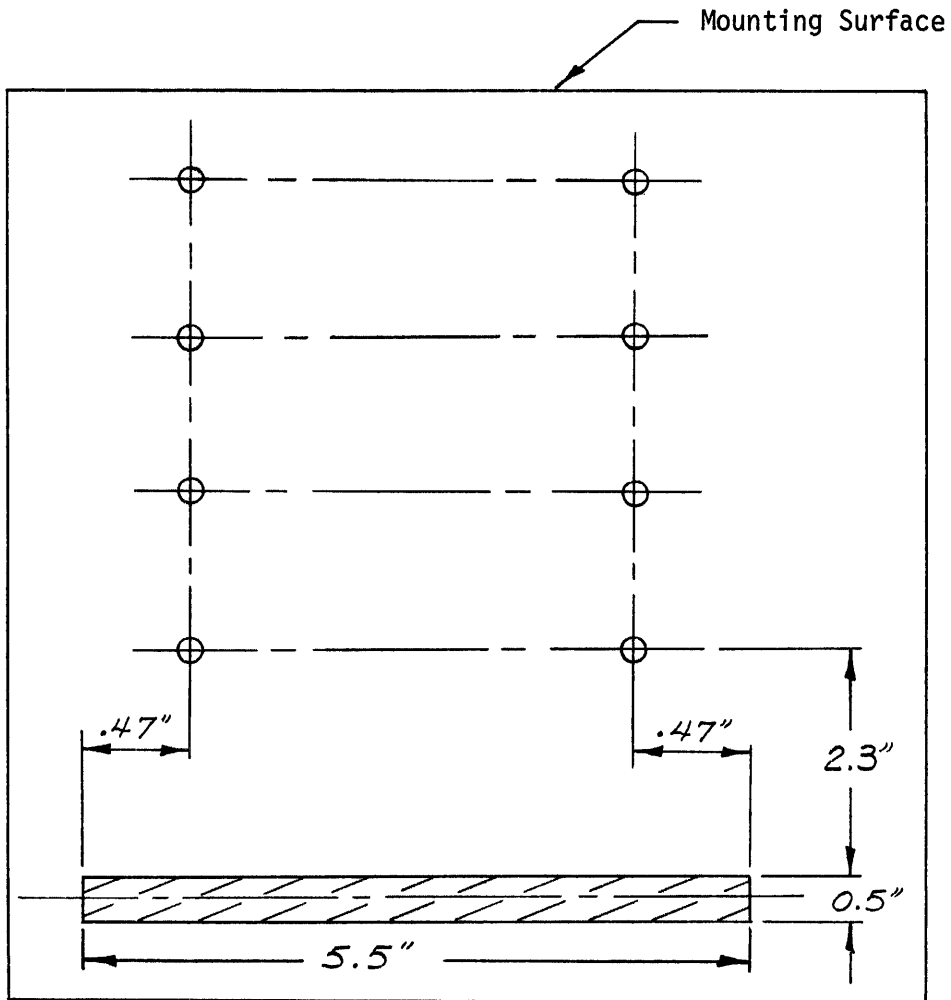




Figure 5 - Hole Pattern For Side-Mounting  
(See Figure 3 For Dimensional Information)

 TITLE <u>PROM Chips Available Under Part #MRV11-AC</u> DISTRIBUTION <u>Only As Needed</u> ORIGINATOR <u>Dave Schanin</u>	NUMBER 027A
	DATE 1 /25 /78
	PRODUCT MRV11-AC
	PAGE <sub>1</sub> OF <sub>1</sub>

Until recently, there was some concern because a customer ordering MRV11-AC PROMs could get PROMs from one of three different vendors -- MMI, Intersil, or Signetics. The problem was caused by the fact that these PROMs had different personality traits; i.e., some blasted a location to a logic "one" and some blasted a location to a logic "zero". This caused customer problems.

The problem has now been resolved so that anyone ordering MRV11-AC PROMs will only receive Signetics 82S131 parts.

 TITLE <u>Extended Memory for the LSI-11</u> DISTRIBUTION <u>Unrestricted</u> ORIGINATOR <u>Ted Semple</u>	NUMBER 028
	DATE 2 / 2 /78
	PRODUCT MSV11-C & MSV11-D
	PAGE <sup>1</sup> OF 12

SCOPE

This Micro Note describes how an LSI-11 user can expand the memory in an LSI-11 system from the normal maximum of 28K words to 100K words. To do this, a customer must manufacture his own control module. The custom module will allow a user to put multiple MSV11-C and/or MSV11-D memories on one LSI-11 bus.

CAUTION: This Micro Note outlines an approach to extend the LSI-11 memory beyond 28K. The approach taken is not compatible with any of the memory management units available for larger PDP-11's. Therefore, it is unlikely that programs written to take advantage of this extended memory approach will migrate without major revision to the next generation processors for the LSI-11 bus.

THEORY OF OPERATION

The extended memory approach explained in this Micro Note is the approach that has been recommended by the LSI-11 Tech Support Group to customers who desire to expand the memory on their LSI-11 system beyond 28K. This approach allows a customer to expand his memory from 28K to a maximum of 100K words by utilizing BDAL 16 and BDAL 17 signal lines on the LSI-11 bus. Both the MSV11-C and the MSV11-D memories have the capability of decoding address lines 16 and 17. The LSI-11 processor does not have the capability of controlling these address lines. The scheme proposed here outlines a way address lines 16 and 17 can be controlled by the program.

The extended memory approach proposed is a paging scheme. A portion of the standard 16-bit LSI-11 address range is mapped to four different pages which fall within a range addressable with 18 address lines. The address translation used is shown graphically by Figure 1 (on page 2). The left-hand side of the figure is the virtual address map. Virtual

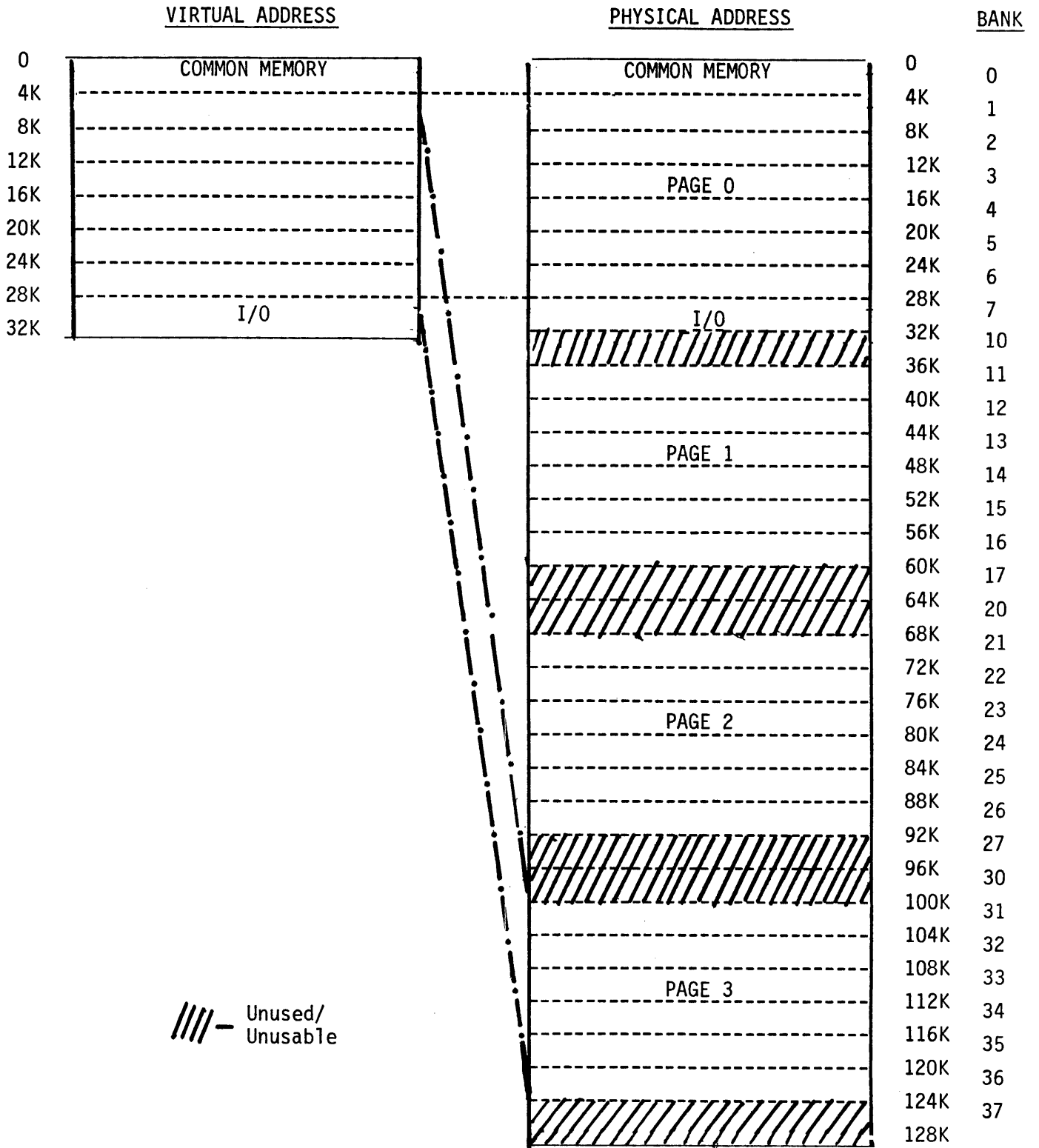


FIGURE #1 - EXTENDED MEMORY ADDRESSING

addresses are 16-bits long and are the actual addresses referred to by LSI-11 or any PDP-11 programs. The right-hand side of the figure shows the physical address map. Virtual addresses 4K to 28K map to physical addresses 4K to 28K for Page 0, 36K to 60K for Page 1, 68K to 92K for Page 2, and 100K to 120K for Page 3.

Two 4K portions of the virtual address range do not map to each of the pages in the physical address page. Both of these 4K ranges are reserved for functions which must be available for programs operating out of any one of the four pages. Virtual addresses 0 to 4K map directly to physical addresses 0 to 4K. This first 4K bank is referred to as Common Memory. The I/O address range 28K to 32K maps directly to physical address range 28K to 32K. Virtual addresses 28K to 32K are not mapped up to 124K to 128K as with memory management units of larger PDP-11's.

Common memory contains all of those system functions which must be accessible independent of which page the program is being executed from. These functions include interrupt vectors, the stack, and all interrupt subroutines. For longer interrupt subroutines, it is possible to have the beginning and the end of the subroutine in common memory with the bulk of the routine actually on one of the pages. Common memory must also be used for routines that handle the switching of the pages. The generation of the physical address, described below, requires that there be a common memory space so that the LSI-11 interrupt mechanism will work properly, regardless of which page a program is being executed from. Any memory reference in the 0 to 4K region will always be in common memory, independent of which page is being used at the time. Additional considerations on the use of common memory are given below.

Figure 2 (on page 4) shows how the physical address is generated. Bits 0 to 15 of the physical address are equivalent to the virtual address. Bits 16 and 17 of the physical address come from the relocation register. The relocation register is a hardware register on a custom module made by the user. The recommended address for the relocation register is 164000.

Figure 3 (on page 4) is the recommended format for the relocation register. Read/write bits 00 and 01 control BDAL 16 and BDAL 17 respectively. Read/write bit 07 is used to enable the bus data address line drivers. Additional circuitry is required on the custom module for the timing of BDAL 16 and BDAL 17 signals. This circuitry is described under the Implementation section below.

Shaded portions of the physical address map (Figure 1) between each one of the pages and the end of memory are defined as unusable. An analysis will reveal that these gaps actually correspond to the virtual address

FIGURE #2 - PHYSICAL ADDRESS FORMAT

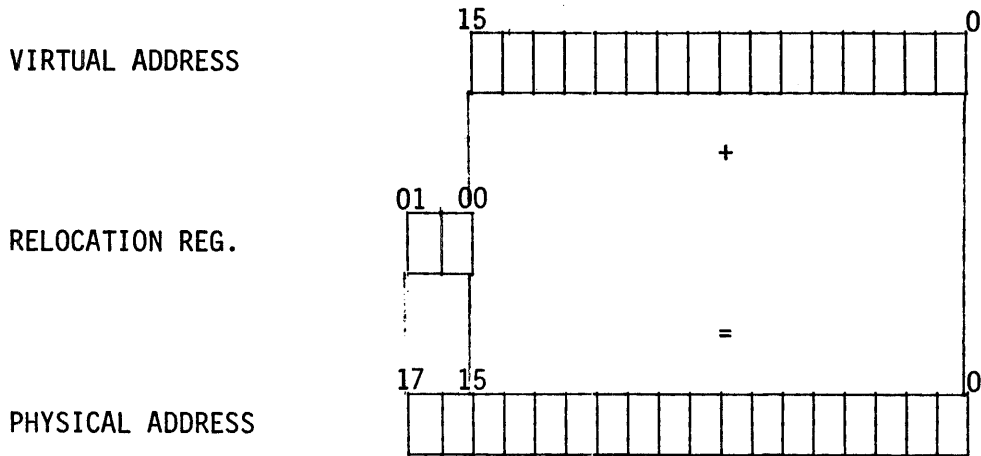
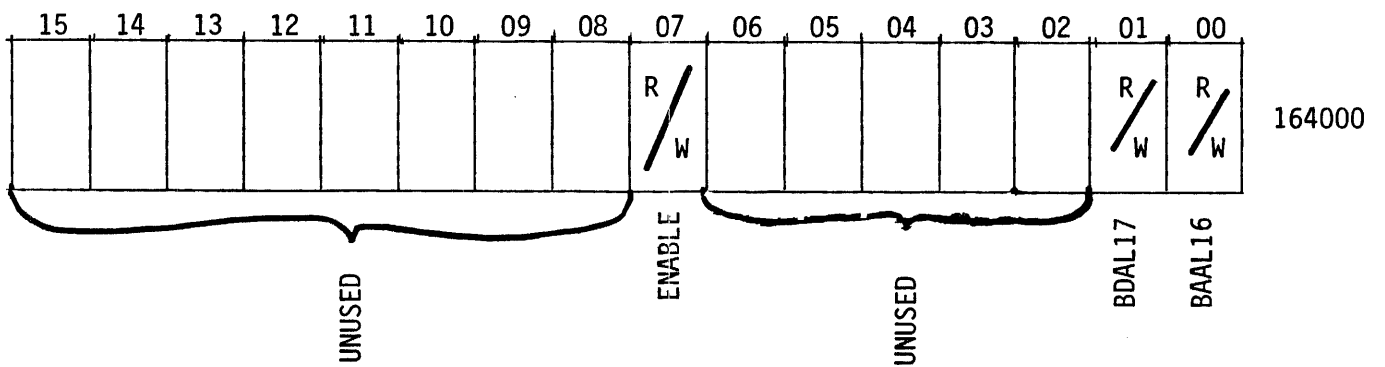


FIGURE #3 - RELOCATION REGISTER





common memory areas and I/O address areas. To make this extended memory addressing scheme work, no memory board can respond to any address within those shaded address ranges of the physical address map. If a board does respond in those ranges, it will mean there are two boards in the memory responding to common memory addresses or I/O addresses. The BBS7 signal on the LSI-11 backplane is generated by the bus master whenever an access to an I/O device is made. This signal inhibits both the MSV11-C and the MSV11-D from responding to any address in the I/O range and from responding to any physical addresses which are memory banks 7, 17, 27, and 37 of the physical address page. Unfortunately, there is no signal similar to BBS7 to prevent the memory boards from responding to addresses in physical address banks 10, 20 and 30 which are translations of the common memory area. Therefore, care must be taken to insure that no memory board responds to addresses in those ranges (a bank 0 detection circuit is described below).

## IMPLEMENTATION

This section describes the physical implementation of how a user can get 100K words of memory on an LSI-11. Configuration of standard LSI-11 memory modules for the extended memory is discussed, and a block diagram for the user's custom module is reviewed.

NOTE: The following approach has been thoroughly reviewed by LSI-11 technical personnel. However, the approach has not actually been breadboarded.

Table 1 (on page 6) shows how standard LSI-11 memory modules are configured for the expanded memory. Either MSV11-D or MSV11-C modules may be used. Switches on either modules are used to select the starting address shown for that particular module. This table shows how to configure the entire 100K words of extra memory. If an application requires less than the full amount of memory, only those memory modules necessary are used.

Figure 4 (on page 8) is the block diagram of the extended memory control module. There are five distinct portions of this block diagram, each with their own functions. These sections are:

1. Bus interface logic
2. Relocation register
3. BDAL drivers
4. Bank 0 detector
5. Synch low stretcher

TABLE #1 - MEMORY MODULE ADDRESS RANGES

<u>PHYSICAL ADDRESS BANK</u>	<u>STARTING ADDRESS</u>	<u>APPROACH #1</u>	<u>APPROACH #2</u>
0	000000	MSV11-CD	MSV11-DD
1			
2			
3			
4	100000	MSV11-CD	MSV11-DD
5			
6			
7			
10	220000	MSV11-CD	MSV11-DD
11			
12			
13			
14	320000	MSV11-CD	MSV11-DD
15			
16			
17			
20	420000	MSV11-CD	MSV11-DD
21			
22			
23			
24	520000	MSV11-CD	MSV11-DD
25			
26			
27			
30	620000	MSV11-CD	MSV11-DD
31			
32			
33			
34	720000	MSV11-CD	MSV11-DD
35			
36			
37			

The following paragraphs discuss the function and operation of each one of these sections.

The bus interface logic performs all of the functions necessary to connect the circuitry of the extended memory control module to the LSI-11 bus. Bus interfacing is accomplished using Digital's own DC004 and DC005 chips. These chips are available from Chipkit DCK11-AA. Four DC005 transceiver chips are used to detect when the extended memory control module is being addressed as well as to buffer the data lines. The DC004 chip handles the LSI-11 bus protocol (the last chip in the kit, the DC003 interrupt protocol chip, is not used). The 8641 bus receiver chip, also available from Digital, buffers the BYSYNC, BSACK, and BINIT signals. Detailed circuit information on how to interface to the bus is available in the DCK11-AA Chipkit User's Guide.

The relocation register is a 3-bit latch (3/4 of a quad latch) with Tri-State buffers to feed the output of the latches back to the data bus. The Tri-State buffers are required to make each bit of the register readable under program control. Control signals from the DC004 bus protocol chip clock data into the latches and enable the Tri-State buffers.

Two 8881 NAND gates, available from Digital, are used for BDAL drivers. One gate drives address data line 16 and the other gate drives address data line 17. Address bits 16 and 17 actually control which page is being used at the time. Data inputs for the drivers come from the relocation register bits 00 and 01. The 8881 gates are enabled by the output of a Schottky negated input AND (NOR) gate. When all inputs to this gate are low, the BDAL drivers are enabled. There are four input signals to this gate. The ENABLE signal from the relocation register permits the programmer to control the bus drivers. The SYNC\* signal comes from the Sync Low Stretcher. This signal enables the extended address bits only during the address portion of a bus transaction. The SACK signal disables the drivers whenever a DMA operation is taking place on the bus. The final input to this gate is BBS0, which comes from the bank 0 detector, disables the 16th and 17th address bits whenever an access is being made to RAM bank 0. A Schottky gate is used to minimize the propagation delay through this circuitry.

The bank 0 detector is a single DC005 transceiver chip. Only the bus receiver and address comparison features of this chip are being utilized. When the address select inputs to the DC005 are allowed to float (vs. being connected either to ground or to Vcc), the chip will detect when BDAL 13, 14, and 15 signal lines are high, bank 0 is being addressed, and the MATCH output of the chip will go high. The MATCH output of the DC005 is the BBS0 signal on the block diagram. When BBS0 is high, the

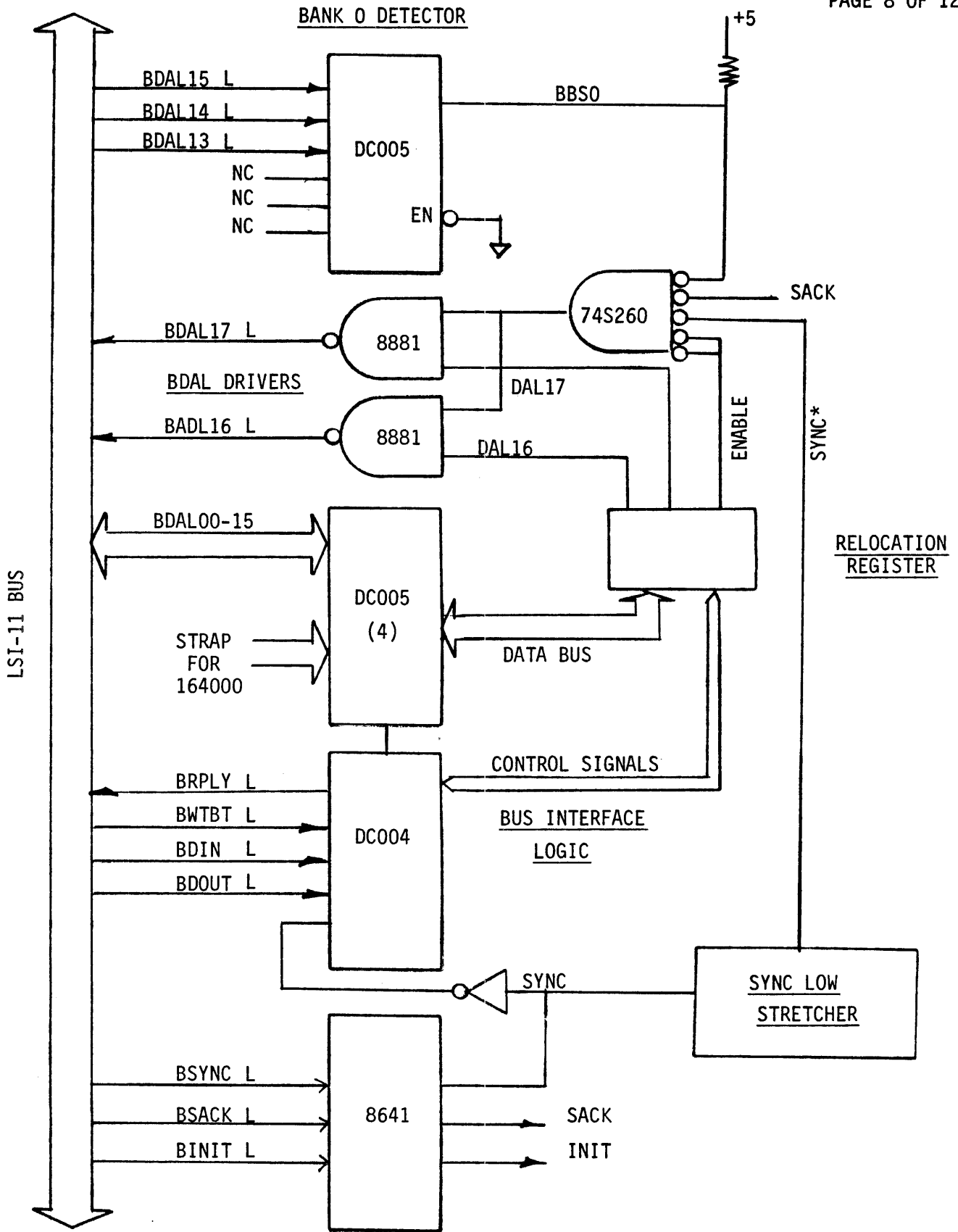


FIGURE #4 - EXTENDED MEMORY CONTROL MODULE BLOCK DIAGRAM

BDAL 16 and 17 drivers will be disabled. The BBS0 signal will not be stable during the data portion of a bus cycle, but that is of no consequence in this design approach.

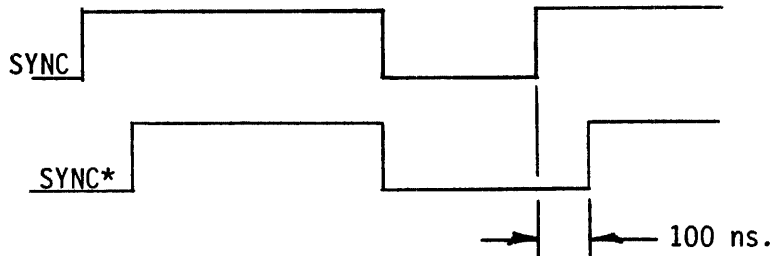


FIGURE #5 - SYNC LOW TIMING DIAGRAM

The final section of the block diagram is the Sync Low Stretcher. As explained above, the function of this circuit is to enable the BDAL drivers only during the address portion of a bus cycle. The LSI-11 bus timing specifications require that the address lines remain stable for 100 ns. after the rising edge of SYNC. The Sync Low Stretcher delays the rising edge of the SYNC signal for 100 ns. so that the timing specification will be met. This circuit may be fabricated from a single flip flop and a one-shot. The one-shot timing should be set for 100 ns. and should be triggered by the rising edge of the SYNC signal. The falling edge of the SYNC signal cannot be used to trigger the one-shot because the time the SYNC signal is low is unpredictable. Figure 5 shows the timing requirements of the Sync Low Stretcher. This circuit makes it possible for the MSV11-D memory module to put parity information onto the bus during the data portion of the bus transaction.

There is an element of risk associated with using the bank 0 detector described above. An analysis of the bus timing circuit diagrams from Chapter 3 of the Microcomputer Handbook will reveal that there is not sufficient amount of time to detect bank 0 and disable the drivers for BDAL 16 and BDAL 17 and still meet the specified address to SYNC set-up time. The bus specification requires that all addresses be stable for at least 75 ns. at the bus receiver before the SYNC signal comes along.

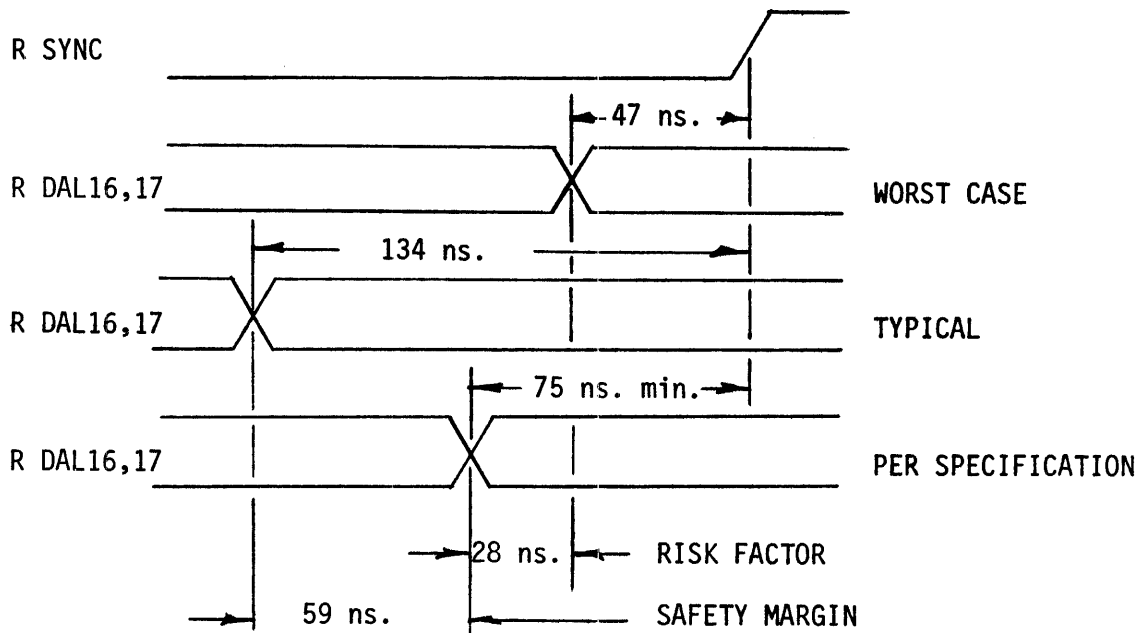


FIGURE #6 - ADDRESS TO SYNC TIMING

Figure #6 shows the timing relationship between the received data address lines 16 and 17 to the received SYNC pulse. The diagram shows worst case timing, typical timing, and the timing required by the bus specification. The worst case occurs when there is the maximum skew between the SYNC signal and the DAL lines on the bus, and there is the maximum propagation delay differential between the drivers and receivers used for the DAL lines and those used for the SYNC lines. Typical bus timing is when there is a minimum amount of bus skewing, and when all of the drivers and receivers used are operating at typical propagation delay times. Normally, the address lines will be stable 59 ns. before the bus specification requires them to be stable. However, when the unlikely worst case condition occurs, the address lines will not be stable until 28 ns. after the bus specification requires them to be stable. The worst case condition is extremely unlikely in systems with short bus lengths as is the case with the new 2x4, 2x8, and 2x12 backplanes. The risk is higher when multiple backplane systems are used and when large, single backplane systems such as the 9x6 backplanes are used. Even in the worst case situation, modules on the bus would have to require stable data for more than 47 ns. before the SYNC signal came along. This, again, is an unlikely situation.

The approach described above is also in violation of the LSI-11 bus loading rules. Using an additional DC005 chip to detect bank 0 results in more than one load on several bus signals. This is not desirable; however, it is required to minimize the bank 0 detection time. The excess loading occurs on non-critical (non edge triggered) bus signals.

## USING COMMON MEMORY

There are several things that a programmer should be aware of when using the paging scheme described above. It was stated that the common memory is required for interrupt vectors, the stack, the beginning and end of all interrupt subroutines, and for the routines where page switching occurs. It is also mandatory that data used by routines on different pages be in common memory. The following paragraphs will elaborate on page switching and interrupt handling.

Page switching operations are required when a routine on one page wants to call or jump to a routine on another page. All page switching must be done from common memory. Coding will be most efficient when a single routine is used for page switching of jumps and another routine for calls.

Figure #7 is an example of a routine that may be used for page switching with calls. Briefly, this routine must store the current page number and program counter so that it is possible to return to the current routine. It must then set the new page number and JSR to the new routine. Upon return from the new routine, the page switching routine must restore the old page number and return to the routine that initially called it. The original routine must specify the new page and virtual address of the new routine. A similar approach may be used for jumping to a new page.

Using this MACRO definition:

```
.MACRO          CALL X PAGE, ADDRESS
JSR             RO, PAGTRN
.WORD          ADDRESS
.WORD          PAGE
.ENDM          CALL X
```


The routine would be:

```
MOV             @#164000, -(SP)
MOV             (RO)+, -(SP)
MOV             (RO)+, @#164000
JSR            PC, @(SP)+
MOV             (SP)+, @#164000
RTS            RO
```

FIGURE #7 --  
PAGE SWITCHING  
ROUTINES FOR CALLS

For the LSI-11 interrupt mechanism to work properly, the interrupt vectors the stack and portions of the interrupt subroutines must be located in the common memory area. For very long interrupt subroutines, it may be desirable to have only the beginning and the end of the routine in the common memory space and the bulk of the routine on one of the extended memory pages. The RTI instruction cannot be executed from one of the pages. Before exiting from the interrupt subroutine, the program must return to common memory. A technique similar to the call page switching routine described above may be used for page switching.



 TITLE <u>Using the MRV11-AA for a Bootstrap ROM</u> DISTRIBUTION <u>Unrestricted</u> ORIGINATOR <u>Ted Semple</u>	NUMBER 029
	DATE 1 / 12 /78
	PRODUCT MRV11-AA
	PAGE1 OF 2

The MRV11-AA can be a cost effective alternative to using the REV11-A bootstrap module in LSI-11/2 systems for customers who use a custom bootstrap ROM. Two features of the REV11-A module are not required with the LSI-11/2 systems. Since the H9281 backplanes have built-in termination networks, the termination networks on the REV11-A are not required. The DMA refresh circuitry on the REV11-A is not required either because the MSV11-D memory module is self-refreshing. Also, the REV11 bootstrap ROMs are now soldered directly to the printed circuit board rather than being mounted in sockets. Customers have discovered that they void their warranty when they remove the ROM and solder in their own custom bootstrap ROM. Using the MRV11-AA is a good alternative to using the REV11 because it is less expensive, has no unneeded features, and the warranty will not be voided.

To maintain standard I/O page address assignments, only 256 x 4 ROMs may be used with the MRV11-AA. Using 512 x 4 ROMs will exceed the 256-word window left at address 173000 in the standard I/O page address assignments. This may or may not be detrimental in a particular application.

One advantage to using the MRV11-AA in place of a modified REV11-A that may not at first be apparent is the MSV11-D feature which allows the memory module to expand its address range from 28K to 30K may now be enabled. The second half of the REV11-A bootstrap ROM has a starting address of 165000 which is in the middle of the 28K to 30K range. This prohibited using the REV11-A in systems that had a MSV11-D module with the expansion feature enabled.

To configure the MRV11-AA as a bootstrap ROM module with starting address 173000, four 256 x 4 ROMs should be inserted in rom CE6, and the straps should be configured as follows:


<u>STRAP NAME</u>	<u>R = REMOVED</u> <u>I = INSTALLED</u>	<u>STRAP NAME</u>	<u>R = REMOVED</u> <u>I = INSTALLED</u>
W0	R	W9	R
W1	R	W10	R
W2	R	W11	I
W3	R	W12	I
W4	R	W13	R
W5	R	W14	I
W6	I	W15	R
W7	R	W16	R
W8	R	W17	R

The QJV11 PROM formatter program may be used to produce the binary patterns for individual PROM chips.

The following table summarizes the SRUN connections made to the 10-pin connector:

BACKPLANE	BACKPLANE PIN CONNECTED TO 10-PIN CONNECTOR
H9270	CH1*
DDV11-B	None--Wire Wrap Pin Available For Customer Use
H9281	AF1
H9273 (11/03-L Backplane)	AF1

**\*NOTE:** Although this backplane used to have a wire wrap connection from CH1 to the 10-pin connector, that wire has been deleted and the connection put in etch. Therefore, a customer using an 11/03, BA11-ME, or an H9270 with a KD11-HA must insert a jumper between pin AF1 and CH1 to obtain the SRUN signal at the 10-pin connector.

 TITLE <u>Extended Bus Time-Out Logic</u> DISTRIBUTION <u>KD11-HA Users</u> ORIGINATOR <u>Joe Austin</u>	NUMBER 032
	DATE 2 / 13 /78
	PRODUCT KD11-HA (M7270)
	PAGE1 OF 3

### SCOPE

This Micro Note discusses the procedures whereby a user can extend the bus time-out delay of the KD11-HA (M7270) LSI-11/2 central processor module using external logic. The normal time-out delay of 12.16 us. is quite adequate for almost all applications. Occasionally, however, a special need arises where the customer wishes to extend this delay. Typical reasons for this could occur as a result of waiting for the response from an exceptionally slow peripheral or allowing time for a complete handshake with another microprocessor system.

Although a modification to the backplane is necessary, no modifications are required to be made to the KD11-HA module.

### THEORY OF OPERATION

Two connections to the bus time-out counter on the KD11-HA module have been routed to the backplane to allow a user direct access to the time-out logic (see Figure 1). The EN/OUT function on E23 pin 7 has been named STOP L (time-out pulse) and is routed to backplane pin AE1. The STB function on E23 pin 10, normally connected to ground to enable the outputs on pins 5 (TERR L) and 6 of E23, has been named MTOE L (time-out enable) and is routed to backplane pin AK1 where it can be connected to pin AL1, which is grounded on the M7270 module. Pins AK1 and AL1 are normally connected together by etch in order to allow for special factory level testing of modules. Thus, when this module is inserted into the backplane, the STB function on E23 is connected to ground and the logic functions normally.

To extend the bus time-out delay beyond the normal 12.16 us., this jumper between pins AK1 and AL1 must be broken so that signal MTOE L can be controlled externally. As long as this signal is kept high, the output signal TERR L (which is used to inform the CPU control chip of the time-out) is blocked. The 6-stage counter contained within the (7497) chip (E23) will continue to run, however, and will count the phase 3 clock (PH 3 H).

Signal STOP L can be used as a clock for driving an external counter as shown in the figure. Based upon a microcycle clock of 380 ns., this signal will be a 380 ns. pulse that will occur after every 64 counts, or 24.32 us. (the normal time-out delay is 32 counts for 12.16 us.).

The figure shows the procedure for interconnecting an additional counter stage to the 7497 chip using a 7474 flip flop. Signal BSYNC insures that the flip flop is reset prior to the start of the bus cycle and that it will be reset again at the end of the cycle. Signal MTOE L will be high at the start of each bus cycle, thus blocking the TERR L line internally to the 7497 via the STB input on pin 10. After the 7497 has counted 24.32 us., it issues the pulse STOP L which sets the flip flop. The STB line to the 7497 then changes to ground, enabling the outputs on E23 pins 5 and 6. The 7497 will then proceed with a normal count-down of 12.16 us. and will generate TERR L to inform the C.P. control chip of the bus time-out. This circuit thus produces a total bus time-out delay of 36.48 us.

By replacing the flip flop with a multiple stage counter, additional bus time-out times can be achieved as shown in the table.

TABLE 1. BUS TIME-OUTS

NUMBER OF COUNTS OF STOP L	NORMAL TIME-OUT (us.)	ADDITIONAL TIME-OUT (us.)	TOTAL TIME-OUT (us.)
0	12.16	0	12.16
1	12.16	24.32	36.48
2	12.16	48.64	60.80
3	12.16	72.96	85.12
4	12.16	97.28	109.44

TOTAL = normal + additional  
= 12.16 us. + 24 us. (no. of counts)

NOTE: Although laboratory tests have verified this design concept, it is provided here for information purposes only. DEC makes no claim and shall not be liable for its accuracy or for the operation of the user's implementation of the design shown.

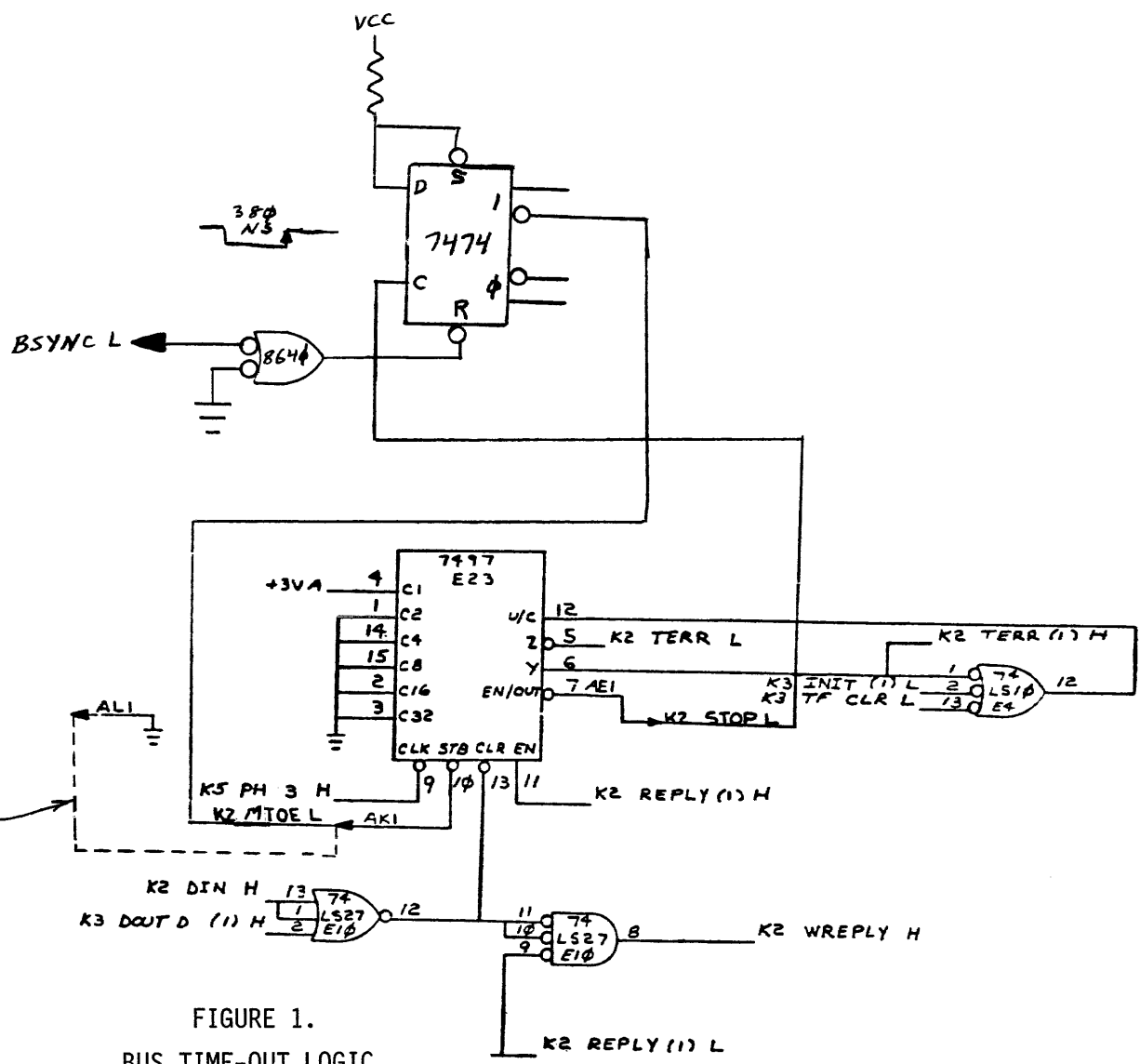



FIGURE 1.  
BUS TIME-OUT LOGIC

digital  
COMPONENTS  
GROUP  
75

REMOVE  
BACKPLANE  
ETCH

	NUMBER 033
	DATE 2 / 24 /78
	PRODUCT DLV11, DLV11-E, -F
	PAGE1 OF 2
TITLE <u>Cables for DLV11, DLV11-E, DLV11-F</u>	
DISTRIBUTION <u>Unrestricted</u>	
ORIGINATOR <u>Lloyd Fugate</u>	

The introduction of the DLV11-EB has raised a question on which cables can be used with "DLV" type asynchronous interfaces. The DLV11-EB is the basic DLV11-E plus BC01V-25 cable. For other DLV interfaces, the BC05C-25 cable has been recommended.

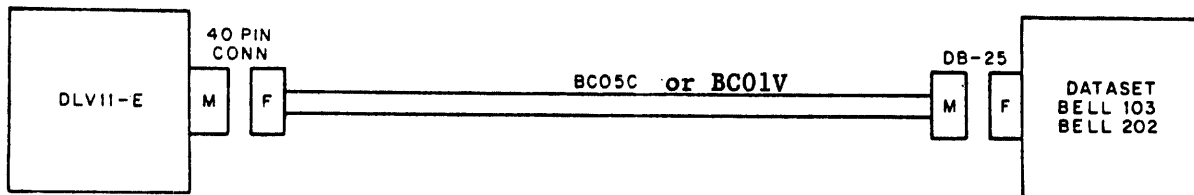
The BC05C-25 is a 40-pin Berg to EIA connector cable that can plug directly into a modem for remote use or to a null modem for local use. This cable contains all 25 specified EIA signals, including those used only for synchronous communication controllers. Since the DLV11 family is asynchronous, almost half of these conductors are not used.

The BC01V-25 is identical to the BC05C-25 but contains only the 15 conductors used for asynchronous communications.

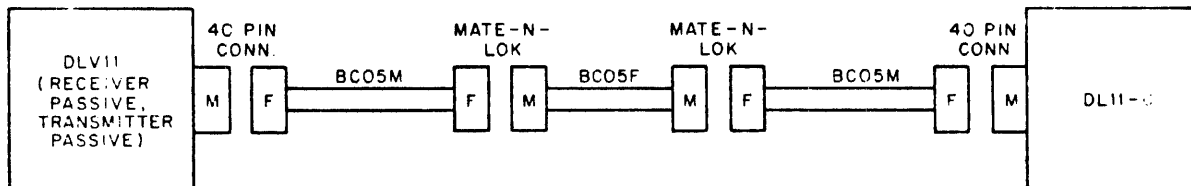
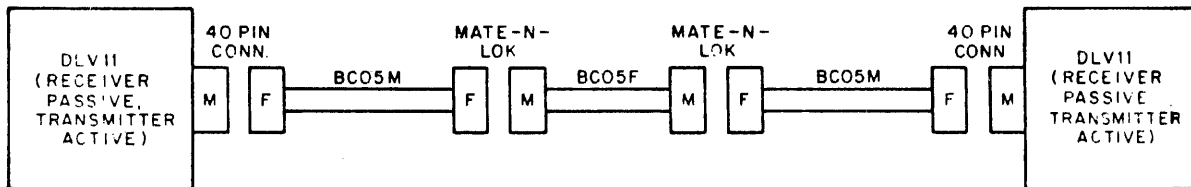
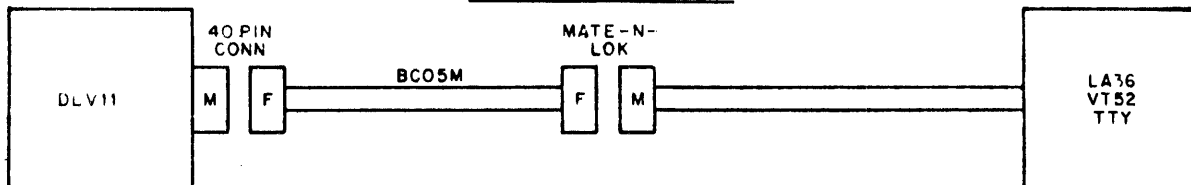
Because of the price difference between these two cables - the BC01V-25 is \$88 and the BC05C-25 is \$112 - future customers will probably use the lower priced cable for their applications. This cable is available in limited quantity now and in volume starting in June.

For those customers who require a cable with null modem capabilities, the BC03M-XX cable is recommended. This cable includes the functionality of the H312 null modem and can be attached to the EIA connector on the BC01V-XX, BC05C-XX, and BC21B-05 (DLV11-J cable).

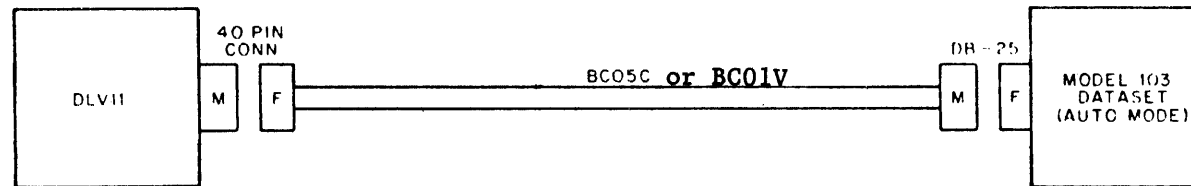
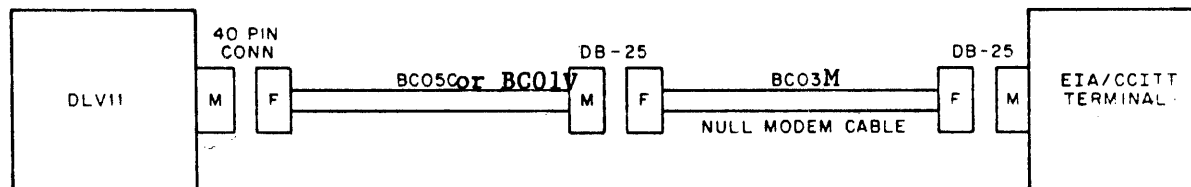
DATA SET CONTROL




CURRENT LOOP MODE



EIA "DATA LEADS ONLY" MODE

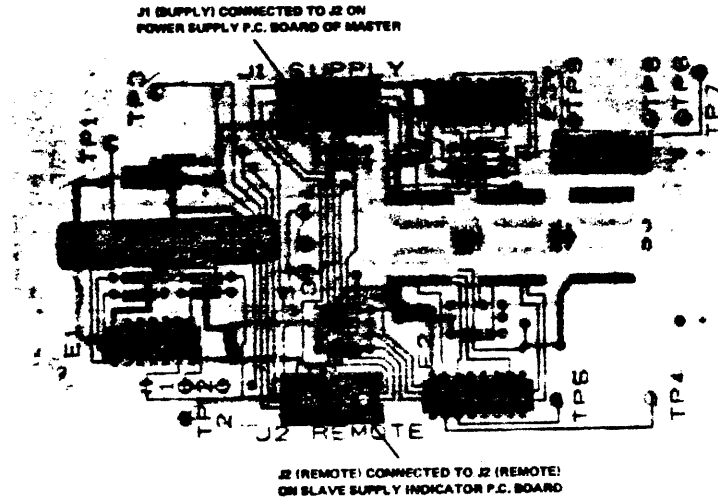




	NUMBER 034
	DATE 3 / 2 78
	PRODUCT 11/03 Systems
	PAGE 1 OF 2
TITLE <u>Configuring a 3-Box 11/03 System</u>	
DISTRIBUTION <u>Unrestricted</u>	
ORIGINATOR <u>Dave Schanin</u>	

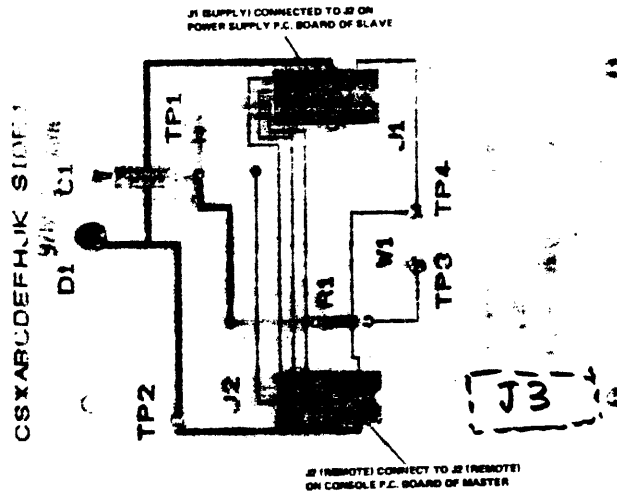
The 11/03 system box and the BA11-ME expander box are both used either stand-alone or in a multiple backplane configuration. When interconnecting boxes in a multiple backplane system, the standard BCV1B or BCV1A cable set interconnects the backplanes. However, the slave power supply in the BA11-ME must be slaved to the master power supply in the 11/03. The front panels on the two boxes have sockets for this purpose (see Figure 1). A cable with a DIP plug on either end is supplied with each BA11-ME to connect J2 on the master to J2 on the slave. The obvious problem is how to connect a second BA11-ME into the system since J2 is now occupied.

The solution, until mid-summer, is to order a BC03Y-16 cable. This cable has three DIP plugs on it, one at either end and one in the center of the cable. One plug goes in J2 of each box. About mid-summer, an ECO will have been phased in, adding J3 to all BA11-ME slave consoles. This plug parallels J2. Therefore, a user could connect J2 on the master 11/03 to J2 on the slave BA11-ME with the cable supplied with the first BA11-ME. J3 on the first BA11-ME can be connected to J2 on the second BA11-ME with the cable supplied with the second BA11-ME. The BC03Y-16 will then no longer be required.




H780-H and -J (Master)

8115-2



H780-K and -L (Slave)

FIGURE 1

	NUMBER 035
	DATE 4 / 4 /78
	PRODUCT MRV11-AA, -BA
	PAGE <sup>1</sup> OF <sup>1</sup>
TITLE <u>PROM Programming</u>	
DISTRIBUTION <u>Unrestricted</u>	
ORIGINATOR <u>Dave Schanin</u>	

INTRO


Customers in a dedicated environment will very often store their program in PROM. Some RAM is always necessary in a PROM-based system for a stack, for a scratchpad, etc. Very often, the RAM on a KD11-F is used, though sometimes a separate memory board is used.

PROBLEM

On power-up, a program must initialize itself to a known state. This includes initializing any scratchpad RAM in the system. However, some PROM customers have assumed a power-up state for the RAMs. On the early 4K RAM parts, this was generally true; they usually powered up to a zero state. However, later version RAMs, which have differential sense amplifiers rather than single ended amps, come up in a random 1 or 0 state. A user who does not initialize the RAM will find that his PROM program will not run unless a certain memory vendor's chips are used in his system because of the power-up state of the RAMs. We cannot, of course, guarantee a particular module will have a particular vendor's chips. It is an obligation of the programmer to clear all RAM on program power-up.

SOLUTION


The software engineer must assure that all RAM in a PROM based system is cleared on program initialization.

	NUMBER 036
	DATE 5 / 16 / 78
	PRODUCT H9273; MMV11-AA
	PAGE 1 OF 1
TITLE <u>Core Memory in 11/03-L Backplane</u>	
DISTRIBUTION <u>Unrestricted</u>	
ORIGINATOR <u>Ted Semple</u>	

THE MMV11-AA CORE MEMORY MODULE MAY NOT BE USED WITH THE 11/03-L BOX, THE BA11-NE EXPANDER BOX, OR THE H9273 BACKPLANE.

The MMV11-AA Core memory module is a quad size printed circuit board. Unlike other quad modules which make all connections to the LSI-11 bus via the A and B card edge connectors, the Core memory module also uses the C and D connectors to interconnect to the LSI-11 bus. The new 11/03-L backplanes do not have the LSI-11 bus distributed on the C and D fingers. Therefore, modules used with the 11/03-L backplane must make all bus connections via the A and B card edge connectors.

To use the MMV11-AA Core memory module in the 11/03-L backplane would require the addition of 10 wires to the backplane to make interconnections to the LSI-11 bus from the C and D connectors and to provide daisy-chain grant continuity on the LSI-11 bus. To do this would be impractical because the backplane used in the 11/03-L box does not have wire-wrappable connector pins. To solder wires onto the backplane would, at best, be a risky operation.

	NUMBER
	037
	DATE
	5 / 30 / 78
TITLE <u>C-D Interconnect Scheme</u>	PRODUCT
DISTRIBUTION <u>Unrestricted</u>	H9273; BA11-N; 11/03-L
ORIGINATOR <u>Joe Austin</u>	PAGE 1 OF 3

### H9273 BACKPLANE

The BA11-N backplane is a 9x4 backplane that accepts both double height and quad height modules. The backplane structure is unique, providing two distinct buses - the LSI-11 bus and the C-D bus as shown in Figure 1. Modules are inserted in rows 1 and 9 of the backplane. The LSI-11 bus signals appear on slots A and B; the C-D bus signals appear on slots C and D.

### LSI-11 BUS SIGNALS

The LSI-11 bus in the H9273 backplane is supplied by slots A and B. These signals are bused to the same pin in all nine rows of the backplane. Interrupt acknowledge and bus grant signals (BIAKO L, BIAKI L, BDGMO L, and BDGMI L) are not bused, being corrected, instead in a way that allows the grant lines to be passed on in a priority chain in order from slot 1 to slot 9, with slot 1 having the highest priority and slot 9 the lowest.

### C-D BUS SIGNALS

The C-D bus is supplied by slots C and D as shown in Figure 2. The +5V supply voltage is bused to all rows on pin A2 of slots C and D (i.e., pins CA2 and DA2). Likewise, ground connections on pins CC2, CT1, DC2, and DT1 are bused to all rows. All other pins connect only to an adjacent row. For example, pin CF2 of any row connects only to pin CF1 of the adjacent higher numbered row. Pins on side 2 of the slot (B2, C2, etc.) connect to the adjacent higher numbered row (except DT2, which connects to CT2 of the adjacent lower numbered row), while pins on side 1 of the slot (B1, C1, etc.) connect to the adjacent lower numbered row (except pin A1, which connects to C1 of the adjacent higher numbered row). Thus, each row, except 1 and 9, has 33 signal connections (i.e., connections other than +5V and ground) to both the adjacent higher numbered row and the adjacent lower numbered row.

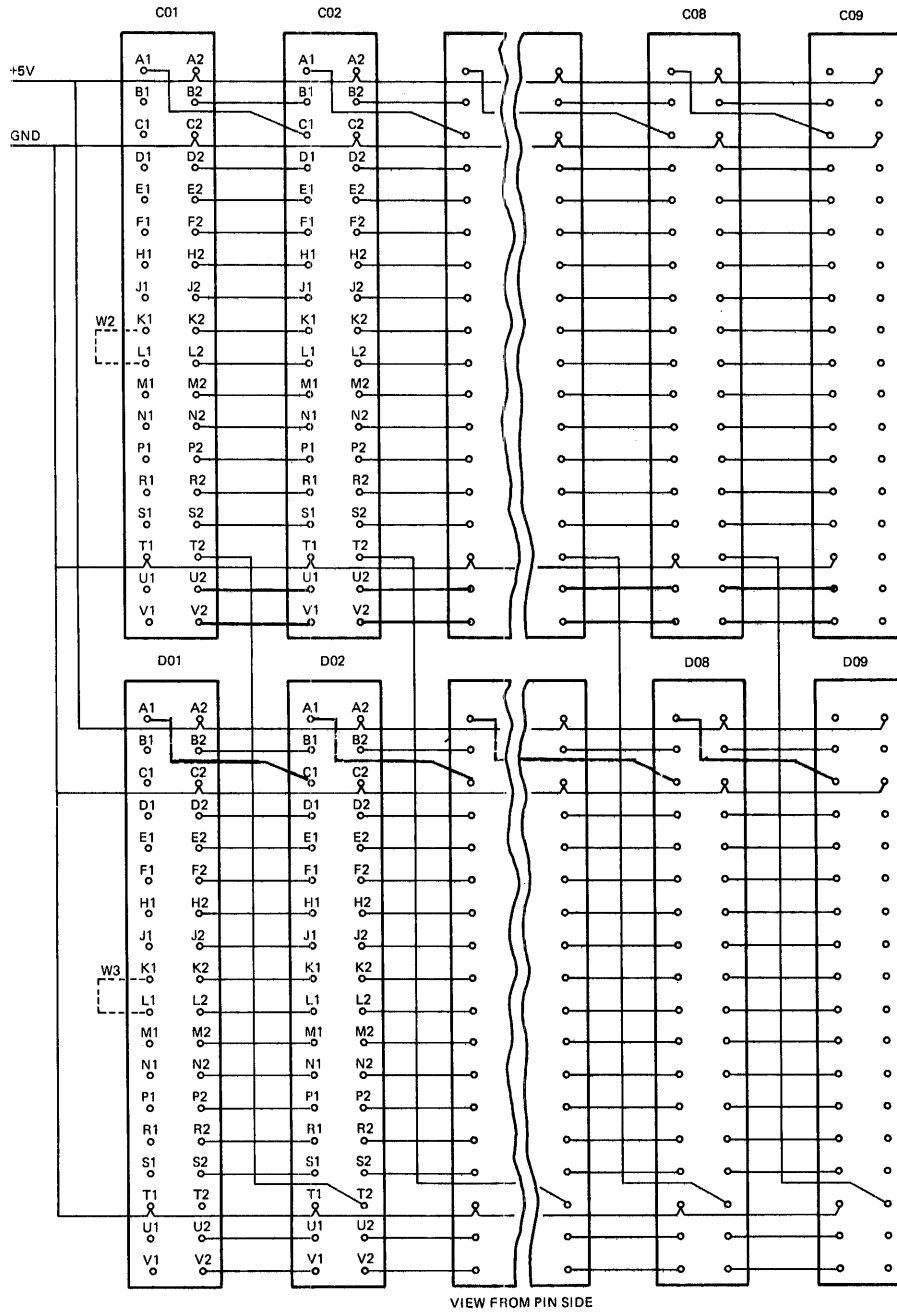
LSI-11 BUS

C-D BUS

	SLOT A	SLOT B	SLOT C	SLOT D
ROW 1	LSI-11 OPTION 1			
ROW 2	LSI-11 OPTION 2			
ROW 3	LSI-11 OPTION 3			
ROW 4	LSI-11 OPTION 4			
ROW 5	LSI-11 OPTION 5			
ROW 6	LSI-11 OPTION 6			
ROW 7	LSI-11 OPTION 7			
ROW 8	LSI-11 OPTION 8			
ROW 9	LSI-11 OPTION 9			

VIEW IS FROM MODULE SIDE OF CONNECTORS.


FIGURE 1 -- H9273 BACKPLANE



- FEATURES:
- ALL PINS A1 CONNECT TO PINS C1 IN THE NEXT LOWEST SLOT.
  - ALL PINS A2 CONNECT TO +5 VOLTS.
  - ALL PINS T2 OF SLOT C ARE CONNECTED TO PIN T2 OF SLOT D IN THE NEXT LOWER SLOT.
  - ALL PINS C2 AND PINS T1 ARE GROUND.
  - JUMPER W2 IS CONNECTED ACROSS PINS K1 AND L1 IN SLOT C ONLY.
  - JUMPER W3 IS CONNECTED ACROSS PINS K1 AND L1 IN SLOT D ONLY.

FIGURE 2 -- C-D BUS WIRING

MR-1564

 TITLE <u>    Diagnostics for 30K Memories on LSI-11's    </u> DISTRIBUTION <u>    Unrestricted    </u> ORIGINATOR <u>    Lloyd Fugate    </u>	NUMBER 038
	DATE 6 / 13 / 78
	PRODUCT MSV11-D
	PAGE 1 OF 1


The recent introduction of the MSV11-D memory system allows for the jumper selection of 2K of memory in the space historically reserved for the I/O page. When this jumper is installed, the LSI-11 is capable of addressing 30K words of memory; however, the installation of this jumper causes some of the LSI-11 diagnostics to exhibit problems.

These problems center around the fact that the first four locations in the I/O page have been used by diagnostic software. The principal use of these locations was in the design of autosizing algorithms and as a means of validating the correct operation of the hardware to conditions of non-existent memory references.

For the MSV11-DD, the memory diagnostic is being upgraded to isolate problems in the 28-30K range; however, there are no plans to modify other diagnostics to support this memory space. Customers should be aware that when a complete system diagnostic is required, they will have to restrap their MSV11-DD memory for 28K if they expect all standard DEC diagnostics to operate.

An effort will be made in the future to document where diagnostics fail as a result of the 30K strap.



	NUMBER 039
	DATE 6 / 27 / 78
	PRODUCT All
	PAGE 1 OF 2
TITLE DMA Request/Grant Timing	
DISTRIBUTION All DMA Users	
ORIGINATOR Ted Semple	

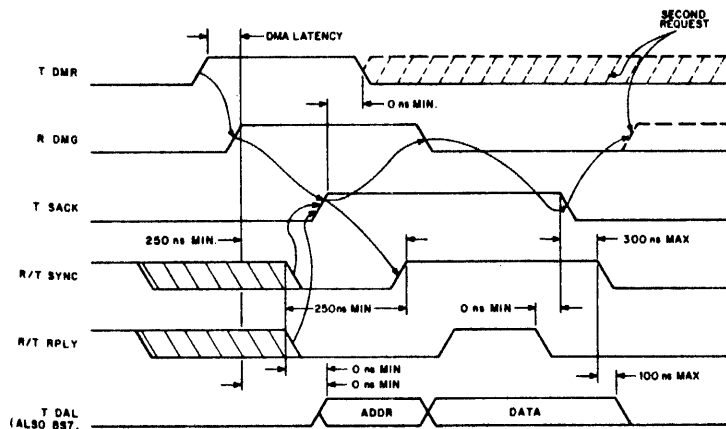
Some LSI-11 users have discovered that their Direct Memory Access modules are not compatible with the KD11-HA processor module, even though they had been successfully used with KD11-F processor modules. There are no compatibility problems with DIGITAL's DRV11-B Direct Memory Access controller or with DIGITAL's DC010 DMA controller chip. Compatibility problems did occur, however, with REV11 modules and with users' custom-designed DMA interfaces. Revision J REV11's have had an ECO incorporated that makes the REV11's DMA refresh compatible with all LSI-11 processors. (Users that desire DMA refresh in systems with a KD11-HA processor must use a Revision J or later version of the REV11 module.)

Both the KD11-F processor and the KD11-HA processor meet the timing requirements for DMA on the LSI-11 bus. There is a difference between processors -- this is the amount of time taken to respond to bus signals. The KD11-F takes longer to issue a Direct Memory Access Grant (DMG) in response to a Direct Memory Access Request (DMR). The KD11-F also takes longer to de-assert DMG after receiving a Select Acknowledge (SACK).

The compatibility problem is caused by the manner in which a Direct Memory Access controller responds to a Direct Memory Access Grant issued by the processor. The LSI-11 bus specification requires that a Direct Memory Access controller meet three criteria before issuing a SACK signal. DMG must be asserted and SYNC and RPLY must be de-asserted before issuing a SACK signal. This requirement is shown by arrows on the timing diagram on Page 2.

Some DMA controllers have interpreted the above timing requirements to mean that a SACK signal can be issued immediately upon the receipt of DMG from the processor and then waiting until SYNC and RPLY are de-asserted before actually starting a bus transaction. The processor module de-asserts DMG upon receipt of a SACK signal from the DMA controller. The apparent failure mode for most DMA controllers is that they require DMG be asserted as well as SYNC and RPLY de-asserted when they start a bus transaction. If DMG is no longer present, then the DMA controller may "hang-up" the bus with SACK asserted. This can happen with either the KD11-F processor or the KD11-HA processor. However, this is more likely to happen with the KD11-HA processor because the KD11-F processor has a longer internal time delay from the receipt of SACK to the de-assertion of DMG.


There are two ways to fix this problem. The first is for the DMA controller to meet the actual bus timing requirements before issuing the SACK signal. The second approach which is not as acceptable but still works with the current LSI-11 processors is to latch the DMG signal on the DMA controller module so that the board does not forget that it received a grant. By latching the DMG signal, the board can then issue SACK immediately upon receipt of the DMG.



- NOTES
1. Timing shown at requesting device bus driver inputs and bus receiver outputs.
  2. Signal name prefixes are defined below:  
T = Bus Driver Input  
R = Bus Receiver Output
  3. Bus Driver Output and Bus Receiver Input signal names include a "B" prefix.

CP-1776

**DMA Request/Grant Timing**

	NUMBER	040
	DATE	6 / 27 / 78
	PRODUCT	BASIC/PTS
	PAGE 1	OF 2
TITLE	Patches for BASIC/PTS on LSI-11	
DISTRIBUTION	Unrestricted	
ORIGINATOR	Rich Billig	

The PDP-11 Paper Tape BASIC software package (BASIC/PTS, QJ900) is not designed to operate on the LSI-11 processor. The patches required for LSI-11 operation are supplied in this Micro Note. Although these patches have been used successfully in a number of installations, they are not "official", and BASIC/PTS is not an officially-supported product on the LSI-11.

The procedure for patching BASIC/PTS is as follows:

1. Load BASIC with the Absolute Loader. The system responds:

```
BASIC V001A
OPT FNS: A-ALL, N-NONE, I-IND
```

2. Press the BREAK key to enter ODT. The system responds:

```
@
```

3. Determine whether you have BASIC/PTS with or without strings. (System responses are underlined for clarity. The character ␣ indicates a carriage return.)

For BASIC/PTS with strings, enter the following patches:

```
@xxxxx
@15160/106746 ␣
@15162/106427 ␣
@15214/106426 ␣
@15222/106426 ␣
@20620/106746 ␣
@20622/106427 ␣
@20672/106426 ␣
@20700/106426 ␣
@P
```

For BASIC/PTS without strings, enter the following patches:

@xxxxx

@13400/106746 2

@13402/106427 2

@13434/106426 2

@13442/106426 2

@16610/106746 2

@16612/106427 2

@16662/106426 2


@16670/106426 2

@P

4. Respond to the question:

OPT FNS: A-ALL, N-NONE, I-IND

DISCLAIMER: Although the patches are known to work, this procedure has not been officially tested.

	NUMBER	041
	DATE	7 / 5 / 78
	PRODUCT	BDV11-AA
	PAGE 1 OF 1	
TITLE	New Functionality for BDV11-AA Boot	
DISTRIBUTION	Unrestricted	
ORIGINATOR	Lloyd Fugate	

Engineering is currently going through a ROM revision to provide new functionality for the BDV11-AA boot. Specifically, this revision adds the following features to the BDV11-AA:


1. Add boot for the RXV21 floppy disk.
2. Code to auto load the full 16KW PROM (2716 type) or 2K PROM (2708 type) resident on the board in a manner which is transparent to the user.
3. Maintain unattended boot but provide option to boot by device name mnemonic.
4. Add DLV11-F DECnet boot.
5. Modify the memory test to support the 30K word MSV11-DD.
6. Add a mechanism so that new boot devices can be added by additional ROMs without change to the operator interface or the basic DEC-supplied ROMs.

These additions do not affect the current functionality of the BDV11.

The revision number for the BDV11-AA may be identified by ROM part numbers as follows:

<u>SOCKET</u>	<u>REV. A PART #</u>	<u>REV. B PART #</u>
E48	23-010E2	23-045E2
E53	23-011E2	23-046E2

It is our plan to phase this revision in during July.

	NUMBER 042
	DATE 7 / 17 / 78
	PRODUCT A11
	PAGE 1 OF 2
TITLE <u>Removing Modules from "Live" Backplanes</u>	
DISTRIBUTION <u>LSI-11 Customers</u>	
ORIGINATOR <u>Joe Austin</u>	

PROBLEM

Occasionally the situation arises where users would like to remove a module from a backplane while the DC bus power is on. The primary reason for doing this is to prevent the loss of programs stored in the MOS memory.

SOLUTION

Our single answer to this is:

!! DON'T !!

DC operating power on the bus should always be turned off whenever a module is being removed or inserted into a backplane. Failure to do so will risk damaging not only the module being changed but also the other modules in the backplane. The inconvenience of a damaged module is far greater than the inconvenience of the loss of contents of MOS memories suffered by turning the backplane power off.

REASONS

- 1) When power is turned off in a system, the DC voltages decay exponentially in a controlled known tested manner. When a module is removed from a live backplane, this controlled decay is bypassed and all power stored on the module must decay as best as it can without the benefit of a ground return line.
- 2) Removing or inserting modules changes the loading on the backplane which causes spikes on data, control, and power lines. Voltages and grounds are not usually removed simultaneously; any combination is possible, including the removal of a ground while +5 and +12 are active. This process, along with spikes that occur as an active line is connected or disconnected, causes random effects, some of which can overload IC's, especially MOS circuits. The probability of a MOS circuit being damaged is high, but the probability of TTL circuits failing under such circumstances is also high.

**digital**  
**COMPONENTS**  
**GROUP**

- 3) If a program is running at the time, it is almost certain that it will malfunction when the module is changed.
- 4) It is almost a certainty that a module will be damaged after fewer than 10 removals or insertions into a "live" backplane.


## ALTERNATIVE PROCEDURE

One method of turning off bus DC power while retaining the memory contents requires the use of techniques similar to providing battery backup for the MSV11-C, D, and E series memory modules. The procedure is to provide a separate power supply similar to a battery backed up supply for the refresh logic of these modules that is separate from the bus power. Detailed procedures for providing battery backup for these memory modules is described in the Application Note entitled "Battery Backup for LSI-11 Microcomputer Systems", document #ED-09371. With the memory modules backed up in this manner, the remainder of the bus is "dead", and the probability of damage being caused to modules removed from the backplane is considerably reduced. The following restrictions still exist, however:

- 1) Memory modules using the alternate power source cannot be removed;
- 2) When removing/inserting modules immediately adjacent to the memory modules, ensure that the module being moved does not make physical contact with the module containing the alternate power. Such contact could conceivably cause a short that would either damage one or more modules or cause loss of memory contents.
- 3) Even though the main bus power is off, some bus signals might be active due to the presence of the refresh power on the memory modules. These signals might be active due to the fact that refresh power is sometimes routed to bus circuits involved in external refresh. Even in a self-refresh configuration, the routing of refresh power throughout the etch of the memory module might leave some bus drivers or receivers active. This has the potential for causing problems as a result of removing or inserting other modules in the backplane, but because of the low power levels, the probability of damage is low.

## DISCLAIMER

The information in this document is furnished for information purposes only. Further development is necessary by the user in order to ensure safe operation. The battery backup procedures defined in the Application Note referenced above have been tested. The removal/insertion of modules under these conditions has not been tested. Consequently, DEC makes no claims and shall not be liable for the accuracy or for the operation of this board removal/insertion procedure.

	NUMBER	043
	DATE	7 / 19 / 78
	PRODUCT	RLV11, RL01
	PAGE 1 OF 2	
TITLE	Backplanes for the RLV11 (RL01)	
DISTRIBUTION	RLV11 Customers and Users	
ORIGINATOR	Joe Austin	

PROBLEM

The control logic for the RLV11 option is contained on two quad modules that need to be interconnected. Signals are brought off these modules on connector fingers in slots C and D to allow for this interconnection via the special etch of the C-D side of the H9273 backplane (Micro Note #037).

QUESTION NO. 1

Can different backplanes be used for the RLV11 disk modules for the LSI-11 and PDP-11/03 systems?

ANSWER NO. 1

No. The only backplane that will support the RLV11 disk modules is the H9273. The H9273 backplane can be purchased by itself; it is also included in the PDP-11/03-L series products and is included in the expansion box, BA11-N.

QUESTION NO. 2

Can any other DEC backplanes containing the LSI-11 bus be modified to accept the RLV11 modules?

ANSWER NO. 2

No. There are no other bused backplanes that will accept these modules and there are no bused backplanes that can be modified to accept them.

The DDV11-B 9x6 backplane, with its uncommitted slots E and F, provides +5V and ground to slots E and F according to standard DEC pin assignments. A conflict exists between one of these standard assignments and a signal on the C-D interconnections between the two RLV11 modules. Because of this conflict, the two RLV11 modules cannot be interconnected by wire-wrapping appropriate pins of the E-F slots of the DDV11-B. The DDV11-B cannot be modified because it is a multi-layer board with +5V and ground in the middle layers.




QUESTION NO. 3

Can the unbused backplane, the H9271, be wire-wrapped to accommodate the RLV11?

ANSWER NO. 3

We recommend against doing this. Wire-wrapping the H9271 backplane represents an extremely high-risk situation to the user which is neither cost-effective nor supported by DEC. The high risk exists because wire-wrapping increases the bus capacitance and crosstalk. This violates the bus specification and decreases significantly the system reliability.

	NUMBER	044
	DATE	7 / 26 / 78
	PRODUCT	MSV11-D, -E
	PAGE 1 OF 1	
TITLE <u>Console ODT "L" Command on 30K Systems</u>		
DISTRIBUTION <u>All MSV11-D Users</u>		
ORIGINATOR <u>Rich Billig</u>		

The Console ODT "L" command, which is normally used to load the paper tape absolute loader, will load into the low 28K words of memory only. Therefore, on 30K word systems, the "L" command will still load the absolute loader just below the 28K word boundary.

The algorithm used by the console micro-program in executing this command is:

1. Starting downwards from address 160000 (8), try to write each word of memory until the highest address location installed responds.
2. Write the CSR address given in the "L" command into this highest RAM location for later use by the absolute loader.
3. Start the loading process.

This memory sizing technique will fail to correctly calculate the highest read/write address if MRV11-BA EPROM board(s) with the reply-to-DATO ECO are installed above the read/write memory in the LSI-11's memory map.

If your application makes non-standard use of the "L" command, please note how the memory sizing algorithm operates in 30K systems.

<b>uNOTE</b>		NUMBER 046A SUPERSEDES uNOTE #046
TITLE DLV11-F Replacement for the DLV11		DATE 12 / 27 / 78
DISTRIBUTION PDP-11/03 and LSI-11 Users		PRODUCT DLV11; DLV11-F
ORIGINATOR Joe Austin		PAGE 1 OF 3

The DLV11-F is almost a direct replacement for the DLV11 and is now being shipped in lieu of the DLV11. The factory configuration of the jumpers on both modules makes them functionally equivalent, even though the jumpers themselves are different.

Forward compatibility (the DLV11-F replacing a DLV11) is maintained in all but three areas:

- (1) The DLV11 uses bit 15 of the receive CSR to indicate dataset status when in the EIA mode; no such bit or function exists on the DLV11-F. (This bit is sometimes used by customers as an operational status bit for terminals as well as modems.)
- (2) When the 20 mA transmitter on the DLV11 is changed from active to passive, the flow of current is also reversed. However, the current flow is not normally reversed when the DLV11-F transmitter is changed to passive. This means that plug compatibility is not maintained with the normal configuration jumpers. Fortunately, this problem can be easily overcome by crossing the configuration jumpers between the transmitter and the connector on the DLV11-F, as shown in Figure 1. Since the configuration jumpers use wire wrap pins, this change is easy to make. (The number of users that will be affected by this is extremely small.)
- (3) The DLV11-F does not accommodate 200 baud.

DLV11-F -- The most significant additional features of the DLV11-F are the separate, programmable baud rates for both the transmit and receive lines; the baud rate increase to 19200 baud; and the additional error status bits.

- (4) The DLV11-F is cleared by INIT, whereas the DLV11 is cleared by DCOK. This difference has a significance only in that the two modules respond differently whenever the system is initialized. This difference usually shows up when the LSI-11 is being down line loaded from a host computer. If the host initializes the bus with a RESET command, then the DLV11-F will generate a different response to the host than will the DLV11.

- (5) The DLV11-F automatically configures 110 baud lines for 2 stop bits and all other baud rates for 1 stop bit. The DLV11 configured the stop bits separately from the baud rate, allowing any combination of the two. It is believed that it is possible to connect the jumpers on the DLV11-F in such a way as to get around this difficulty. If so, it will be discussed in a later Micro Note.

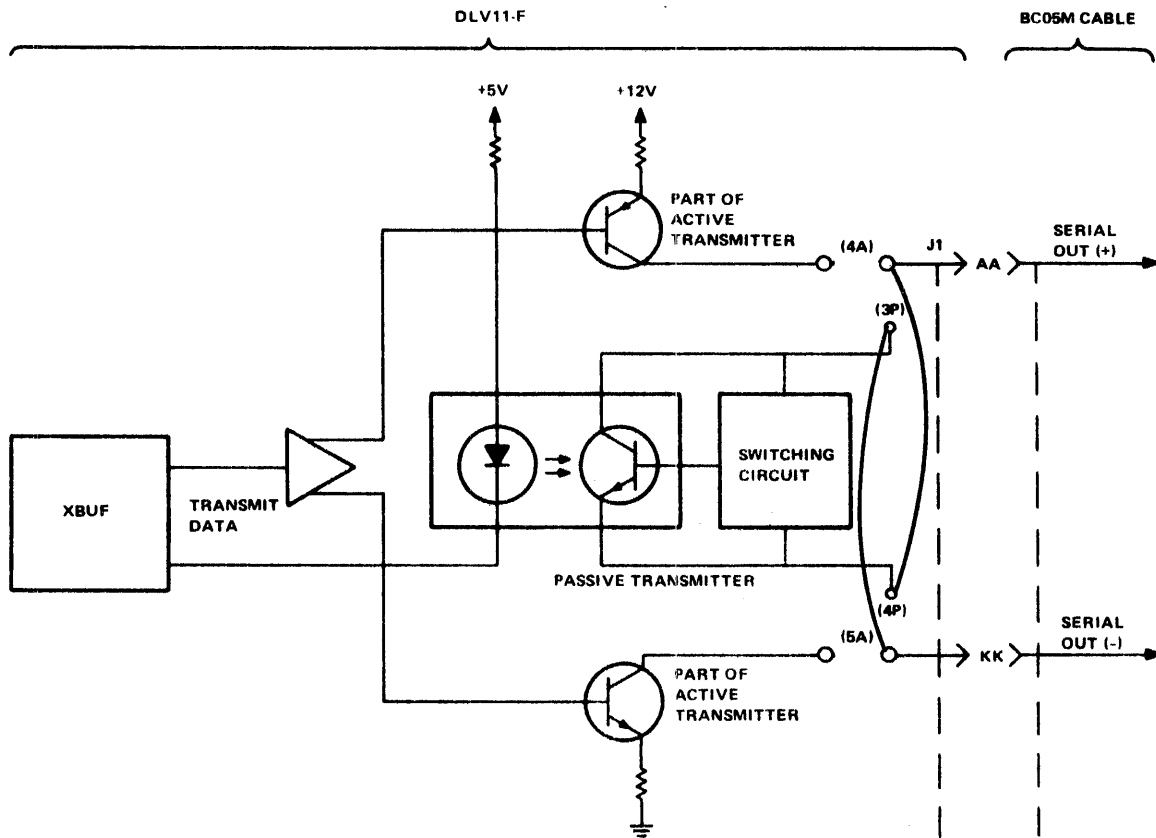



FIGURE 1 -- DLV11-F TRANSMITTER CONFIGURED FOR PASSIVE  
20 mA OPERATION AND DLV11-COMPATIBLE REVERSED  
CURRENT FLOW

	NUMBER 047
	DATE 12 / 19 / 78
	PRODUCT KDF11, REV11
	PAGE 1 OF 2
TITLE <u>Incompatibility Between the REV11 and the LSI-11/23</u>	
DISTRIBUTION <u>Unrestricted</u>	
ORIGINATOR <u>Joe Austin</u>	

PROBLEM

The ROM bootstrap on the REV11-A and -C will not work with the LSI-11/23 processor (KDF11). Any attempts by the LSI-11/23 processor to run this bootstrap will cause the system to fail.

SOLUTION

In all cases where the REV11 is installed in a backplane containing the LSI-11/23 processor, the bootstrap on the REV11 must be disabled. This is accomplished by removing W4 from the REV11 module (M9400-YA, -YC).

REASONS FOR FAILURE

Two programming bugs exist in the REV11 ROM that are incompatible with the KDF11.

The first occurs in the memory address test at location 173712 where the following illegal addressing mode combination is used: MOV R2, (R2)+. This instruction is executed differently in the LSI-11/23 than it is in the LSI-11.

The second bug occurs at location 165326 in the RX01 bootstrap. In this case, a write byte instruction is used where a write word instruction should be used. The write byte instruction assumes that bit 14, which is used by the RX01 as an initialize command, is a zero. With the LSI-11 processor, bit 14 is zero; with the LSI-11/23, it is not.

REMAINING USEFUL FUNCTIONS

With the ROM disabled, the REV11 can be used as a terminator in LSI-11/23 systems, and it can be used as a refresh source for memory modules that do not have internal refresh.

ALTERNATE BOOTSTRAPS

Bootstraps are provided for several peripherals on the following modules that are compatible with the LSI-11/23:

MXV11 (with the MXV11-A2 ROM set)

RLV11 (RL01)  
RXV11 (RX01)  
RXV21 (RX02)  
RKV11 (RK05) (See Note 1)  
TU58

BDV11-AA (with the -045 and -046 ROMs)

RKV11 (RK05) (See Note 1)  
RLV11 (RL01)  
RXV11 (RX01)  
RXV21 (RX02)  
DECnet

NOTE 1: The RKV11 cannot be used in LSI-11/23 systems using 18-bit addressing; i.e., with greater than 64KB of memory.

<b>μnote</b>	NUMBER	048
	DATE	12 / 20 / 78
	PRODUCT	KDF11-A
	PAGE 1 OF 10	
TITLE	LSI-11/23 Instruction Timing	
DISTRIBUTION	Unrestricted	
ORIGINATOR	Ted Semple, Doug Swartz & Burt Hashizume	

This Micro Note consists of four sections:

- 1.0 -- Instruction Execution Time--Standard PDP-11 & EIS
- 2.0 -- Floating Point Instruction Timing
- 3.0 -- DMA Latency
- 4.0 -- Interrupt Latency

#### 1.0 -- LSI-11/23 INSTRUCTION EXECUTION TIME--STANDARD PDP-11 & EIS

The execution time for an instruction depends on the instruction itself, the modes of addressing used, and the type of memory referenced. In most cases, the instruction execution time is the sum of a Basic Time, a Source Address (SRC) Time, and a Destination Address (DST) Time.

$$\text{INSTR TIME} = \text{Basic Time} + \text{SRC Time} + \text{DST Time}$$

$$(\text{BASIC TIME} = \text{Fetch Time} + \text{Decode Time} + \text{Execute Time})$$

Some of the instructions require only some of these times.

The tables that follow are typical (typ. gate delays, max. MSV11 access time, refresh overhead, and 40 ns. bus propagation delays) instruction execution times for standard memories. A 300 ns. microcycle time is assumed. Times can vary +20%. All timing is in microseconds unless otherwise noted.

If memory management is enabled, add .16 us. for each memory reference. To arrive at incremental value to add to the instruction times given in the tables, use the following (select numbers from the memory cycle column):

$$\text{Increment} = .16 \text{ (reads and writes)} + .32 \text{ (read/modify/write)}$$



TABLE #1 -- BASIC TIMES

Instructions	Micro-Cycles	Memory Cycles			MSV11-C	MSV11-D	MSV11-E
		R	W	RMW			
MOV, CMP, BIT, ADD, SUB, BIC, BIS, XOR, SXT, CLR, TST, SWAB, COM, INC, DEC, NEG, ADC, SBC, ROR, ROL, ASR, ASL, MFPS	4	1			1.85 us.	1.72 us.	1.76 us.
MTPS	14	1			4.85 us.	4.72 us.	4.76 us.
MFPI (D)	12	1			4.25 us.	4.12 us.	4.16 us.
MTPI (D)	6	2			3.10 us.	2.85 us.	2.93 us.
MUL	78-80	1			24.65 us.	24.52 us.	24.56 us.
DIV	132-167	1			50.75 us.	50.62 us.	50.66 us.
ASH	14-101	1			30.95 us.	30.83 us.	30.86 us.
ASHC	21-155	1			47.15 us.	47.02 us.	47.06 us.
All Branch Instructions	4	1			1.85 us.	1.72 us.	1.76 us.
SOB (Branch)	7	1			2.75 us.	2.62 us.	2.66 us.
(No Branch)	6	1			2.45 us.	2.32 us.	2.36 us.
RTS	7	2			3.40 us.	3.15 us.	3.23 us.
MARK	12	2			4.90 us.	4.65 us.	4.73 us.
RTI, RTT	12	3			5.55 us.	5.17 us.	5.29 us.
Set or Clear C,V,N,Z	7	1			2.75 us.	2.62 us.	2.66 us.
HALT	21-25	3	2				
WAIT	8	1			3.05 us.	2.92 us.	2.96 us.
RESET	8-430	1			124.70 us.	124.57 us.	124.61 us.

TABLE #1 -- BASIC TIMES (CONTINUED)

Instructions	Micro-Cycles	Memory Cycles			MSV11-C	MSV11-D	MSV11-E
		R	W	RMW			
EMT, TRAP	17	3	2		9.11 us.	7.95 us.	8.07 us.
IOT, BPT	20	3	2		10.01 us.	8.85 us.	8.97 us.
JMP (mode 1)	5	1			2.15 us.	2.02 us.	2.06 us.
(mode 2)	6	1			2.45 us.	2.32 us.	2.36 us.
(mode 3)	6	2			3.10 us.	2.85 us.	2.93 us.
(mode 4)	6	1			2.45 us.	2.32 us.	2.36 us.
(mode 5)	7	2			3.40 us.	3.15 us.	3.23 us.
(mode 6)	7	2			3.40 us.	3.15 us.	3.23 us.
(mode 7)	9	3			4.65 us.	4.27 us.	4.39 us.
JSR (mode 1)	9	1	1		4.38 us.	3.86 us.	3.90 us.
(mode 2)	10	1	1		4.68 us.	4.16 us.	4.20 us.
(mode 3)	10	2	1		5.33 us.	4.69 us.	4.77 us.
(mode 4)	10	1	1		4.68 us.	4.16 us.	4.20 us.
(mode 5)	11	2	1		5.63 us.	4.99 us.	5.07 us.
(mode 6)	11	2	1		5.63 us.	4.99 us.	5.07 us.
(mode 7)	13	3	1		6.88 us.	6.11 us.	6.23 us.

TABLE #2 -- SOURCE TIMES

Source Addressing (mode 0)	0			0	0	0
(mode 1)	2	1		1.25 us.	1.12 us.	1.16 us.
(mode 2)	2	1		1.25 us.	1.12 us.	1.16 us.
(mode 3)	4	2		2.50 us.	2.25 us.	2.33 us.
(mode 4)	3	1		1.55 us.	1.42 us.	1.46 us.
(mode 5)	5	2		2.80 us.	2.55 us.	2.63 us.
(mode 6)	5	2		2.80 us.	2.55 us.	2.63 us.
(mode 7)	7	3		4.05 us.	3.67 us.	3.79 us.

TABLE #3 -- DESTINATION ADDRESSING

Instructions	Micro-Cycles	Memory Cycles					
		R	W	RMW	MSV11-C	MSV11-D	MSV11-E
MOV, CLR, (mode 0)	0				0	0	0
SXT, MFPS, (mode 1)	4		1		2.23 us.	1.84 us.	1.84 us.
MTPI (D) (mode 2)	4		1		2.23 us.	1.84 us.	1.84 us.
(mode 3)	5	1	1		3.18 us.	2.66 us.	2.70 us.
(mode 4)	4		1		2.23 us.	1.84 us.	1.84 us.
(mode 5)	6	1	1		3.48 us.	2.96 us.	3.00 us.
(mode 6)	6	1	1		3.48 us.	2.96 us.	3.00 us.
(mode 7)	8	2	1		4.73 us.	4.09 us.	4.17 us.
CMP, BIT, (mode 0)	0				0	0	0
TST (mode 1)	3	1			1.55 us.	1.42 us.	1.46 us.
(mode 2)	3	1			1.55 us.	1.42 us.	1.46 us.
(mode 3)	4	2			2.50 us.	2.25 us.	2.33 us.
(mode 4)	3	1			1.55 us.	1.42 us.	1.46 us.
(mode 5)	5	2			2.80 us.	2.55 us.	2.63 us.
(mode 6)	5	2			2.80 us.	2.55 us.	2.63 us.
(mode 7)	7	3			4.05 us.	3.67 us.	3.79 us.
MTPS, MFPI (D), (mode 0)	0				0	0	0
MUL, DIV, ASH, (mode 1)	-1	1			0.35 us.	0.22 us.	0.26 us.
ASHC (mode 2)	-1	1			0.35 us.	0.22 us.	0.26 us.
(mode 3)	0	2			1.30 us.	1.05 us.	1.13 us.
(mode 4)	-1	1			0.35 us.	0.22 us.	0.26 us.
(mode 5)	1	2			1.60 us.	1.35 us.	1.43 us.
(mode 6)	1	2			1.60 us.	1.35 us.	1.43 us.
(mode 7)	3	3			2.85 us.	2.47 us.	2.59 us.
BIC, BIS, (mode 0)	0				0	0	0
ADD, SUB, (mode 1)	5			1	3.18 us.	2.66 us.	2.70 us.
SWAB, COM, (mode 2)	5			1	3.18 us.	2.66 us.	2.70 us.
INC, DEC, (mode 3)	6	1		1	4.13 us.	3.49 us.	3.57 us.
NEG, ADC, (mode 4)	5			1	3.18 us.	2.66 us.	2.70 us.
SBC, ROR, (mode 5)	7	1		1	4.43 us.	3.79 us.	3.87 us.
ROL, ASR, (mode 6)	7	1		1	4.43 us.	3.79 us.	3.87 us.
ASL, XOR (mode 7)	9	2		1	5.68 us.	4.91 us.	5.03 us.

2.0 -- FLOATING POINT INSTRUCTION TIMING

The execution time of a KEF11-A floating point instruction is dependent on the following:

1. type of instruction
2. type of addressing mode specified
3. type of memory

In addition to the above, the execution time of many instructions, such as ADDF, are dependent on the data.

Table 1 provides the basic instruction times for addressing mode 0 with a microcycle time of 300 ns. Tables 2 through 5 show the additional time required, using the MSV11-E, for instructions with other than mode 0. Refer to the notes for the execution time variations for the data dependent instructions.

TABLE 1

<u>Instruction</u>	<u>Micro-cycles</u>	<u>Mode 0 Time (us.)</u>	<u>Notes</u>	<u>Modes 1 thru 7</u>
LDF	28	9.15	1,2,19	Use Table 2
LDD	36	11.55	1,2,23	
LDCFD	40	12.75	1,4	
LDCDF	55	17.25	1,5	
CMPF	65	20.25	14,15	
CMPD	71	22.05	14,15	
DIVF	301	91.05	1,29,41,43,44	
DIVD	795	239.25	1,30,42,43,44	
ADDF	121	37.05	1,16,17,18,20,25,27,28,41,43,44	
ADDD	139	42.45	1,16,21,22,24,26,27,28,42,43,44	
SUBF	124	37.95	1,16,17,18,20,25,27,28,41,43,44	
SUBD	142	43.35	1,16,21,22,24,26,27,28,42,43,44	
MULF	264	79.95	1,29,31,41,43,44	
MULD	641	193.05	1,30,32,42,43,44	
MODF	682	205.35	1,26,30,32,33,34,35,41,43,44	
MODD	693	208.65	1,26,30,32,33,34,36,42,43,44	
TSTF	28	9.15	1,2,37	
TSTD	32	10.35	1,2,37	

TABLE 1 (CONTINUED)

<u>Instruction</u>	<u>Micro-cycles</u>	<u>Mode 0 Time (us.)</u>	<u>Notes</u>	<u>Modes 1 thru 7</u>
STF	18	6.15		Use Table 3
STD	26	8.55		
STCDF	65	20.25	1,38	
STCFD	48	15.15	1,4	
CLRF	36	11.55		
CLRD	40	12.75		
<hr/>				
ABSF	43	13.65	37	Use Both Tables 2 and 3
ABSD	51	16.05	37	
NEGF	42	13.35	1,37	
NEGD	50	15.75	1,37	
<hr/>				
LDFPS	11	4.05		Use Table 4
LDEXP	38	12.75	1,2,3,37	
LDCIF	60	18.75	6,8	
LDCID	55	17.25	6,8	
LDCLF	60	18.75	6,7,8,9	
LDCLD	55	17.25	6,7,8,9	
<hr/>				
STFPS	16	5.55		Use Table 5
STST	17	5.85		
STEXP	34	10.95	1,2	
STCFI	58	18.15	11,12,39	
STCDI	59	18.45	11,12,39	
STCFL	55	17.25	10,11,13,40	
STCDL	56	17.55	10,11,13,40	
<hr/>				
CFCC	12	4.35		No Operands
SETF	14	4.95		
SETD	14	4.95		
SETI	14	4.95		
SETL	14	4.95		

TABLE 2

Addressing Mode	Microcycles*		Read/Write Memory Cycles		Time (us.)	
	Single Precision	Double Precision	Single	Double	Single Precision	Double Precision
	1	6,8,11	8,10,13	2/0	4/0	4.81
2	7,9,11	9,11,14	2/0	4/0	5.11	7.22
2 Immediate	6,8,11	2,4,7	1/0	1/0	4.05	2.85
3	7,9,11	9,11,14	3/0	5/0	5.86	7.97
4	7,9,11	9,11,14	2/0	4/0	5.11	7.22
5	8,10,13	10,12,15	3/0	5/0	6.16	8.27
6	8,10,13	10,12,15	3/0	5/0	6.16	8.27
7	10,12,15	12,14,17	4/0	6/0	7.52	9.63

\*Note: The three numbers (of microcycles) in each set represent three different conditions:

1. if the floating point number is positive
2. if the floating point number is negative and non-zero
3. if the floating point number is a negative zero with FIUV flag clear

TABLE 3

Addressing Mode	Microcycles		Read/Write Memory Cycles		Time (us.)	
	Single Precision	Double Precision	Single	Double	Single Precision	Double Precision
	1	3	5	0/2	0/4	2.56
2	6	8	0/2	0/4	3.46	5.72
2 Immediate	-2	-6	0/1	0/1	0.23	-0.97
3	4	6	1/2	1/4	3.61	5.87
4	6	8	0/2	0/4	3.46	5.72
5	5	7	1/2	1/4	3.91	6.17
6	5	7	1/2	1/4	3.91	6.17
7	7	9	2/2	2/4	5.27	7.53

TABLE 4

Addressing Mode	Microcycles		Read/Write Memory Cycles		Time (us.)	
	Short Integer	Long Integer	Short	Long	Short Integer	Long Integer
1	2	4	1/0	2/0	1.35	2.71
2	3	5	1/0	2/0	1.65	3.01
2 Immediate	1	1	1/0	1/0	1.05	1.05
3	3	5	2/0	3/0	2.41	3.76
4	3	5	1/0	2/0	1.65	3.01
5	4	6	2/0	3/0	2.71	4.06
6	4	6	2/0	3/0	2.71	4.06
7	6	8	3/0	4/0	4.06	5.42

TABLE 5

Addressing Mode	Microcycles		Read/Write Memory Cycles		Time (us.)	
	Short Integer	Long Integer	Short	Long	Short Integer	Long Integer
1	2	4	0/1	0/2	1.43	2.86
2	3	5	0/1	0/2	1.73	3.16
2 Immediate	1	1	0/1	0/1	1.13	1.13
3	3	5	1/1	1/2	2.48	3.91
4	3	5	0/1	0/2	1.73	3.16
5	4	6	1/1	1/2	2.78	4.21
6	4	6	1/1	1/2	2.78	4.21
7	6	8	2/1	2/2	4.13	5.57

## NOTES

1. Add 300 ns. if result is positive.
2. Add 300 ns. if result is non-zero.
3. Add 900 ns. if SRC > 177 or SRC < -177.
4. Add 900 ns. if floating point number = 0.
5. Add 3.3 us. if overflow on rounding.
6. Add 300 ns. if integer is negative.
7. Add 1.5 us. if absolute value of integer < 65,536.

8. Add 1.2 us. n times where  $n = 220 - \text{expn}$ .
9. Add 1.2 us. n times where  $n = 240 - \text{expn}$  and if absolute value of integer  $\geq 65,536$ .
10. Add 600 ns. if  $\text{expn} < 20$ .
11. Add 2.1 us. n times where n is the smaller of the two absolute values:  $(210 - \text{expn})$  or  $(230 - \text{expn})$ .
12. Add 600 ns. if integer is negative.
13. Add 900 ns. if integer is negative.
14. Add 1.2 us. if floating point numbers are equal.
15. Add 2.1 us. if numbers are unequal but the signs are the same.
16. Add 600 ns. if  $\text{FPACC} > \text{FPSRC}$ .
17. Add 2.4 us. if  $\text{FPSRC} > \text{FPACC}$ .
18. Add 600 ns. if adding opposite signs or subtracting like signs.
19. Add 2.4 us. if trapped on undefined variable.
20. Add 900 ns. and 1.2 us. n times where  $n = \text{expn}$  difference.
21. Add 3.6 us. if  $\text{FPSRC} > \text{FPACC}$ .
22. Add 1.2 us. if adding opposite signs or subtracting like signs.
23. Add 1.2 us. if trapped on undefined variable.
24. Add 900 ns. and 1.8 us. n times where  $n = \text{expn}$  difference.
25. Add 1.2 us. n times where  $n = \text{shifts}$  to normalize.
26. Add 1.8 us. n times where  $n = \text{shifts}$  to normalize.
27. Add 3.3 us. if underflow.
28. Add 600 ns. if overflow.
29. Add 600 ns. if need to normalize after multiply or divide.
30. Add 1.2 us. if need to normalize after multiply or divide.
31. Add 600 ns. for every "1" bit in multiplier (FPSRC).
32. Add 1.2 us. for every "1" bit in multiplier (FPSRC).
33. Add 900 ns. times n where  $n = \text{expn} \bmod 16$  (calc integer and fraction).
34. Add 300, 600, or 900 ns. if  $\text{expn} = 21-40, 41-60$  or  $> 100$ , or 61-100 respectively.
35. Add 1.8 us. if the fractional part = 0.
36. Add 1.2 us. if the fractional part = 0.
37. Add 4.5 us. if trapped on any of the FP11 interrupts.
38. Add 5.4 us. if trapped on overflow.
39. Add 24.3 us. if trapped on conversion error.
40. Add 24.9 us. if trapped on conversion error.
41. Add 1.2 us. if rounding.
42. Add 1.8 us. if rounding.
43. Add 8.1 us. if trapped on overflow.
44. Add 9.0 us. if trapped on underflow.





3.0 -- DMA LATENCY

DMA (Direct Memory Access) latency, which is the time from request (BDMRL) to bus mastership for the first DMA device, is:

	<u>IMA LATENCY</u>
MSV11-C	4.15 us., worst case
MSV11-D	3.49 us., worst case
MSV11-E	3.53 us., worst case

Worst Case Time = Longest DATIO Cycle + Refresh Time

A 300 ns. microcycle time is assumed.

4.0 -- INTERRUPT LATENCY

Interrupt latency is the sum of the time from request (BIRQL) to acknowledgement (BIAKL) and the time from acknowledge to fetch of the first service routine instruction.

		<u>BIRQL TO BIAKL</u>	<u>BIAKL TO FETCH</u>	<u>TOTAL WORST CASE</u>
Standard PDP-11 Instruction Set	MSV11-C	12.11 us.	9.41 us.	21.52 us.
	MSV11-D	10.79 us.	8.18 us.	18.97 us.
	MSV11-E	11.07 us.	8.26 us.	19.33 us.
EIS	MSV11-C	56.28 us.	9.41 us.	65.69 us.
	MSV11-D	55.65 us.	8.18 us.	63.83 us.
	MSV11-E	55.81 us.	8.26 us.	64.07 us.
FP11 (KEF Option)	MSV11-C	N/A	N/A	N/A
	MSV11-D	N/A	N/A	N/A
	MSV11-E	47.72 us.	8.26 us.	55.98 us.

For all floating point instructions except ADD, SUB, MUL, DIV and MOD, the interrupt latency is the length of the instruction. Unlike the KD11 processors, the EIS instructions are not interruptable. In the floating point arithmetic instructions, interrupts may be serviced while in the midst of their execution. If an interrupt is to be serviced before execution is complete, the instruction is aborted and all the PDP-11 general registers and floating point registers are restored to their original values. After the interrupt is serviced, the floating point instruction is restarted from scratch. This interrupt restore routine takes 6.9 us. and the time must be added to the interrupt latency times where execution of an instruction is aborted.

<b>ynote</b>		NUMBER 049
TITLE <u>System Differences - LSI-11 vs. LSI-11/23</u>		DATE 12 / 22 / 78
DISTRIBUTION <u>Unrestricted</u>		PRODUCT <u>KDF11-A, KD11-F, KD11-HA</u>
ORIGINATOR <u>Ted Semple &amp; Doug Swartz</u>		PAGE 1 OF 2

Here is a list of system differences between a KDF11-A and KD11-F. It is intended to point out all possible problems that may arise if a KD11-F is literally removed from a backplane and a KDF11-A substituted.


1. KDF11-A HAS NO BOOT LOADER IN MICROCODE -- KD11-F has "L" command. Those users who are down line loading to KDF11-A's will have to change their host software to enter the bootstrap loader via micro ODT, which is 14 memory words. KDF11-A users whose memory size varies will have to self-size the system via micro ODT or enter a quick PDP-11 program to do the same thing. The LSI-11's boot loader automatically sizes memory.

Those users still using paper tape and thus invoking the LSI-11 boot from a terminal will have to enter the 14-word PDP-11 boot by hand from the terminal.

2. KDF11-A WILL NOT PERFORM MEMORY REFRESH, KD11-F DOES -- The dual LSI-11 board (KD11-HA) does not perform refresh either, but nevertheless, there will be some users who pull out a KD11-F, insert a KDF11-A, and then must do something else to keep memory refreshed such as change over to the newer memories (e.g., MSV11-C, MSV11-D) that perform refresh locally.
3. EVENT LINE IS ON LEVEL 6 IN KDF11-A, ON LEVEL 4 IN KD11-F -- With the KDF11-A having the optional capability to support 4 interrupt levels, the real time clock on normal PDP-11 systems is attached to Level 6. In the KD11-F, it is attached to Level 4. Users who have written software and have locked out the event line by setting the priority level to 4 will still see the event line interrupt with the KDF11-A. Users will have to set the priority level to 6 or above to lock out the event line.

DEC software is unaffected since it takes advantage of the fact in the KD11-F that the other 2 bits of the priority level are mechanized (read/write) but do not do anything. In many cases, users do not lock out the clock at all because they do not want to miss a tick, and if they have, the change is trivial and should be only in a small number of places in their code.

4. KDF11-A DOES NOT BRING FOUR EXTRA MICROCODE BITS TO MODULE CONNECTOR AS KD11-F DOES -- The KDF11-A does not have 4 microcode bits like the KD11-F has. Users who are sensing these 4 bits for one reason or another (e.g., bus initialize, bus error, etc.) will have to make a change in their system.
5. The KDF11-A pulses SRUN (AF1, AH1) during ODT each time a character is transmitted. The KD11-F and KD11-HA modules do not pulse SRUN during ODT. All three modules do pulse the SRUN line each time an instruction is fetched. Those users who use SRUN for other than driving the RUN lamp should be aware that they will get additional pulses.
6. The KDF11-A supports 18-bit addressing, whereas the KD11-F and KD11-HA modules only support 16-bit addressing. When the KDF11-A has the MMU enabled, all modules on the bus must be capable of responding to 18-bit addressing. Memory modules must decode all 18 bits. Modules with DMA capability must address 18 bits instead of only 16 bits. I/O interfaces that use BBS7 instead of decoding address bits 13 and above will work with either 16- or 18-bit addressing.
7. There are some differences in instruction execution between the KD11's and the KDF11-A. See Micro Note #053 for details.

	NUMBER 050
	DATE 12 / 29 / 78
	PRODUCT KDF11-A, KD11-F, KD11-HA
	PAGE 1 OF 3
TITLE <u>Micro ODT Differences - LSI-11 vs. LSI-11/23</u>	
DISTRIBUTION <u>Unrestricted</u>	
ORIGINATOR <u>Ted Semple</u>	

The attached micro ODT difference list shows that there are some changes in ODT between the KD11 CPU's and the KDF11-A CPU. Most notably, the KDF11-A does not support the "L" command.

In most cases, those using ODT from a console terminal will not be affected. However, the slight differences in response to some commands may impact users that have programmed a host computer to emulate a console terminal for the purpose of down line loading (DLL) programs to the LSI-11.

MICRO ODT DIFFERENCES: LSI-11 vs. LSI-11/23

KD11-F & KD11-HA

KDF11-A

1. All characters that are input are echoed except when in the APT command mode where no characters are echoed. An echoed line feed (LF) will be followed by a carriage return (CR) only (no second "LF" or padding nulls). This method creates a potential timing problem with a TTY ASR33 which types the next character before the print head has completely returned.
2. When an address location is open, another location can be opened without explicitly closing the first location; e.g., 1000/123456 2000/054321.
3. "↑" will open the previous location.
4. "@" will open a location using indirect addressing.
5. "←" will open a location using relative addressing.
6. "M" will print the contents of an internal CPU register.
7. Rubout (ASCII 177) will delete the last character typed in.
8. "L" is the boot loader command which will load the absolute loader.
9. Control-Shift-S command mode (ASCII 23) accepts 2 bytes forming a 16-bit address and dumps 10 bytes in binary format. The 2 input bytes are not echoed.

All characters that are input in any command mode except the APT mode are echoed except the octal codes 0, 2, 10, 12, 200, 202, 210 and 212. This suppresses echoing LF's (and nulls (0), STX's (2), BS's (10)) because an automatic (CR) and (LF) follow. In the APT command mode, no input characters are echoed.

An address location must be explicitly closed by a (CR) or (LF) command before another is opened or else an error (?) will occur and any open location will automatically be closed without altering its contents.

"↑" is illegal and uODT prints "?", (CR), (LF), "@"

"@" is illegal and uODT prints "?", (CR), (LF), "@"

"←" is illegal and uODT prints "?", (CR), (LF), "@"

"M" is illegal and uODT prints "?", (CR), (LF), "@"

Rubout is illegal and uODT prints "?", (CR), (LF), "@"

"L" is illegal and uODT prints "?", (CR), (LF), "@"

Control-Shift-S command mode (ASCII 23) accepts 2 bytes forming an 18-bit address with bits (17:16) always zeroes and dumps 10 bytes in binary format. The 2 input bytes are not echoed.

KD11-F & KD11-HA (cont.)

10. Up to a 16-bit address and 16-bit data may be entered. Leading zeroes are assigned.
11. Incrementing (LF), the address 177776 results in the address 000000.
12. Incrementing a PDP-11 register from R7 prints out "R8" and the contents of R0.
13. The I/O page is in the address group 17XXXX.
14. The micro ODT mode can be entered from the following sources:
- a) A PDP-11 HALT instruction.
  - b) A double bus error.
  - c) An asserted HALT line.
  - d) A power up option.
  - e) An asserted HALT line caused by a DLV11 framing error.
  - f) A micro ODT bus error.
  - g) A memory refresh bus error.
  - h) An interrupt vector time out.
  - i) A non-existent micro PC address.
15. A carriage return (CR) is echoed and followed by just a line feed (LF).
16. No "H" command.

KDF11-A (cont.)

- Up to an 18-bit address and 18-bit data may be entered. Leading zeroes are assumed.
- Incrementing (LF), the addresses 177776, 377776, 577776 and 777776 result in the addresses 000000, 200000, 400000 and 600000 respectively; i.e., the upper 2 bits of the 18-bit address are not affected. They must be explicitly set.
- Incrementing a PDP-11 register from R7 prints out "R0" and the contents of R0.
- The I/O page is in the address group 77XXXX where address bits (17:12) must be explicit ones.
- The micro ODT mode can be entered from the following sources:
- a) A PDP-11 HALT instruction when in kernel mode; the POKL line is low and the HALT jumper option strap is present.
  - b) An asserted HALT line.
  - c) A power up option.
  - d) An asserted HALT line caused by a DLV11 framing error.
  - e) A micro ODT bus error.
- A carriage (CR) return is echoed and followed by another (CR) and line feed (LF).
- The F-11 chip set has a feature which is not implemented on the KDF11-A. "H" causes the chip set to output a signal that can be used to toggle the HALT F/F to allow single-step program execution. "H" causes the KDF11-A to execute a microcode routine that, in effect, does nothing.

<b>μnote</b>		NUMBER 051
TITLE <u>Digital Supported PROMs</u>		DATE 12 / 22 / 78
DISTRIBUTION <u>Unrestricted</u>		PRODUCT <u>MRV11-AA MRV11-B, MRV11-C MXV11-A, PB11</u>
ORIGINATOR <u>Ted Semple</u>		PAGE 1 OF 1

The following PROMs are supported by Digital to the extent shown below:

UV PROMs	Size	MRV11-B	MRV11-C	MXV11-A	PB11
Intel 2758	8K		X	X	
Intel 2708	8K	X			X
Intel 2716	16K		X	X	X
Intel 2732	32K		X	X	X
Mostek MK2716	16K		X	X	X
T.I. TMS 2516	16K		X	X	X
T.I. TMS 2532	32K		X	X	

Bipolar PROMs	Size	MRV11-AA	MRV11-C	MXV11-A	PB11
Intel 3628	8K		X	X	
Signetics 82S 129	1K	X			X
Signetics 82S 131	2K	X			X
Signetics 82S 2708	8K		X	X	
Signetics 82S 181	8K		X	X	X
Signetics 82S 191	16K		X	X	X

This is not an exhausting list of possible ROM chips. It does represent the chips that have been tested by Digital except for the 32K parts which will be supported when available from the vendors.

The 1758 UV PROM is supported but is not recommended for new designs because continued availability is not guaranteed.

<b>μnote</b>		NUMBER 052
TITLE <u>Parity Memory in LSI-11/23 Systems</u>		DATE 12 / 27 / 78
DISTRIBUTION <u>Unrestricted</u>		PRODUCT KDF11-A MSV11-E
ORIGINATOR <u>Ted Semple</u>		PAGE 1 OF 8

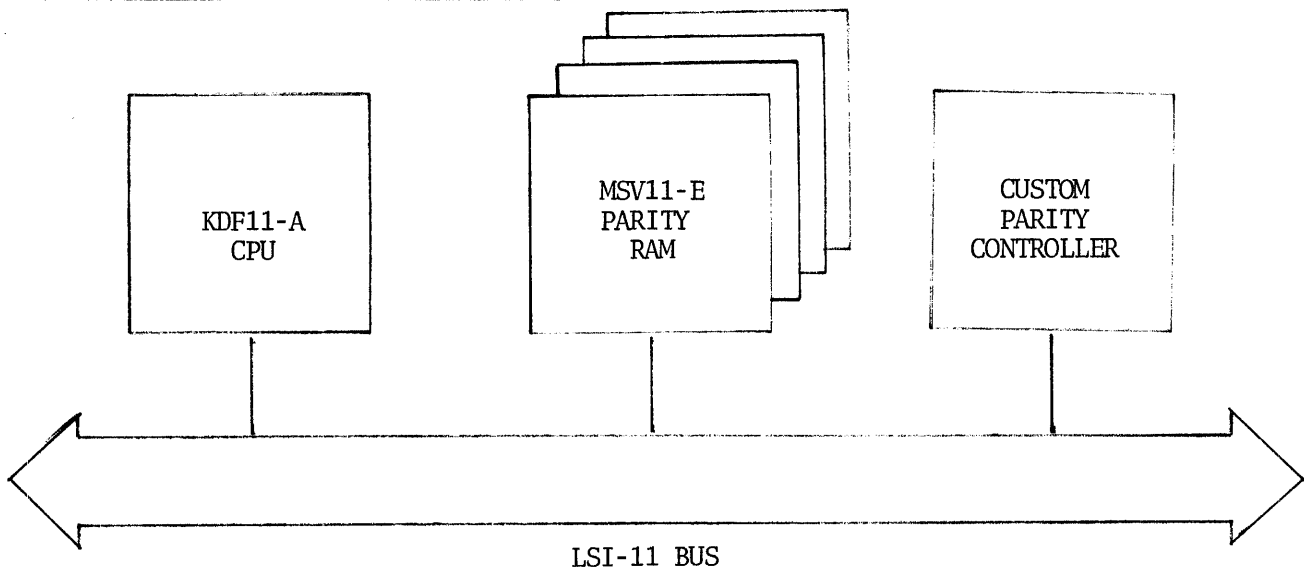


FIGURE 1 -- A CUSTOM PARITY CONTROLLER ALLOWS THE MSV11-E PARITY FEATURE TO BE USED WITH THE KDF11-A

Both the KDF11-A CPU module and the MSV11-E parity memory module have provision to support parity of memory data. However, the parity feature cannot be used unless a parity controller is available. Currently, DIGITAL does not have a parity control module. A customer desiring parity in a LSI-11/23 system must fabricate his own custom parity control module.

This Micro Note explains the parity features of the KDF11-A and MSV11-E modules. An approach for implementing a parity controller is described.



### KDF11-A PARITY MECHANISM

A simple parity trap mechanism is implemented on the module. During every processor-initiated DATI or the DATI portion of a DATIO(B) cycle, the processor will sample the state of bus signals BDAL16L and BDAL17L after the 200 ns. REPLY deskew time. BDAL16L and BDAL17L (which are address bits 16 and 17, respectively, during the beginning of a bus cycle) are treated the same way as BDAL (15:0) at REPLY time. BDAL16L at REPLY time is interpreted as a parity error signal from memory while BDAL17L at REPLY time is interpreted as a parity error enable signal from an external parity controller module. If BDAL17L is not asserted at REPLY time, no abort will occur and program execution will continue. If BDAL17L and BDAL16L are both asserted at REPLY time, a trap to location 114<sub>8</sub> will occur.

The above capability allows the processor to recognize memory parity errors and trap to the same PDP-11 compatible location. Unlike PDP-11 parity controllers which have a Control and Status register that captures the high order seven address bits of the offending location, the KDF11-A will have no information available to the programmer. This information can be captured by external bus logic described below.

### MSV11-E PARITY LOGIC

MSV11-E parity logic functions are shown in Figure 2. The basic functions include a parity generator for memory write data, a 2-bit memory array, a parity detector, and a latch circuit. The parity generator produces two parity bits, one for each main memory byte. Address and control signal (not shown) lines are identical to those applied to the main memory array. When any main memory location is read, the corresponding parity memory location is read. Parity bits are checked by the parity detector and the result is stored in the output latch. If a parity error is detected (PAR ERROR IND H is active), PAR ERR H is gated onto the BDAL16L bus line. The processor reads the error during the memory read (DATI) cycle and responds accordingly.

Parity logic functions are tested when running memory diagnostics by writing incorrect parity bits. The custom parity control module forces this condition during a DATO(B) cycle by asserting BDAL16L during the output data transfer portion of the bus cycle. BDAL16L is received, inverted to produce "write wrong parity" (WWPA16H), and applied to the parity generator, forcing it to store incorrect parity bits. The error is then detected by subsequent memory read cycles.

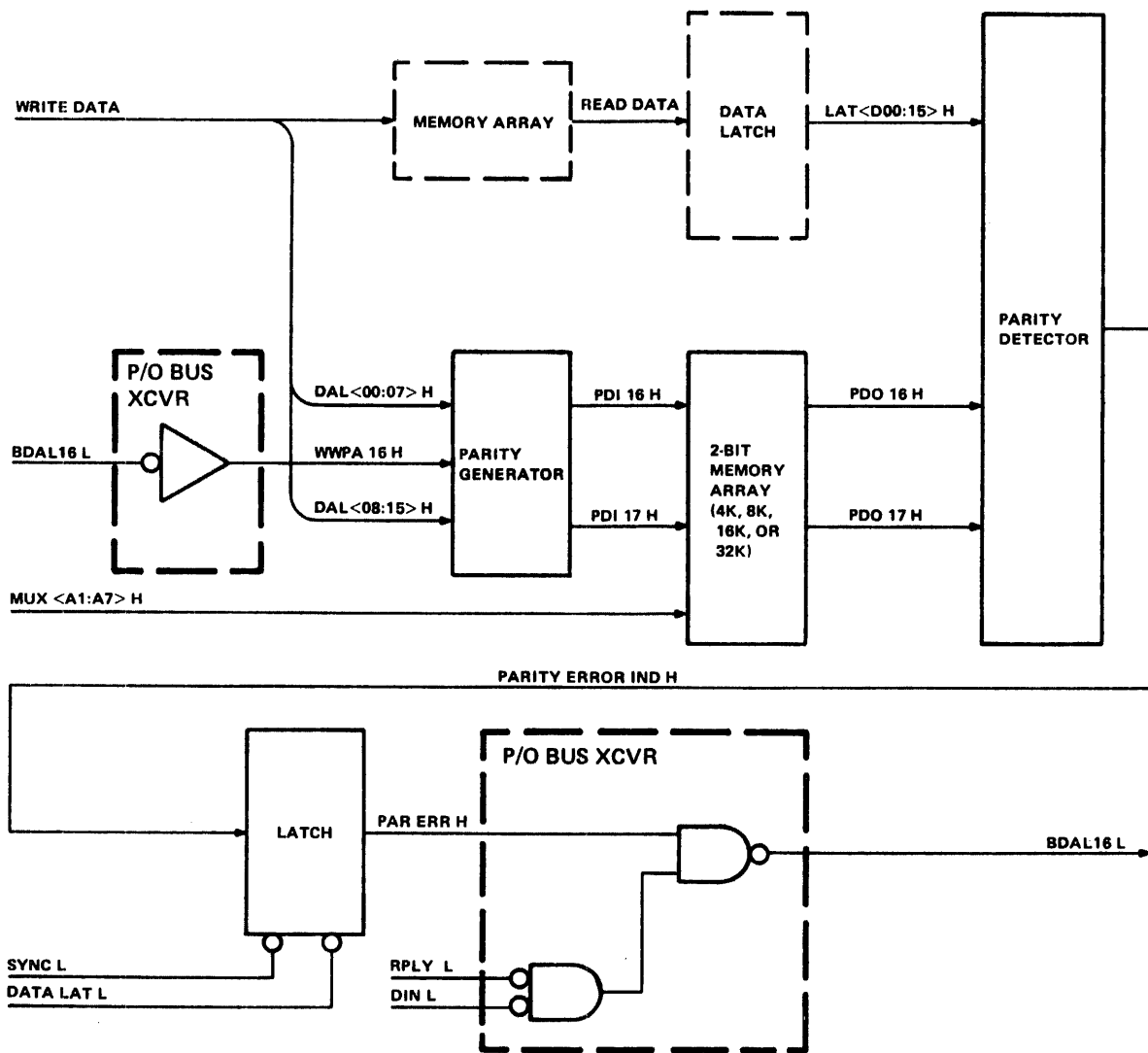


FIGURE 2 -- MSV11-E PARITY LOGIC

### CUSTOM PARITY CONTROL MODULE

To achieve PDP-11 software system compatibility and to properly interface with the KDF11-A and the MSV11-E modules, the parity controller must:

- 1) Have a compatible CSR (Control and Status Register). In addition to parity control functions, the CSR contains the partial address of the memory location with a parity error.
- 2) Have provision to control BDAL17, the parity enable signal.
- 3) Have provision to control BDAL16. BDAL16 forces the MSV11-E to write wrong parity for diagnostic purposes.

### CONTROL AND STATUS REGISTER (CSR)

The CSR can be thought of as one 16-bit register which has its own location address. The contents of the CSR can be either read (DATI) or changed (DATO) by the processor. The CSR contents are also changed when a parity error has occurred. The CSR bit assignments are illustrated in Figure 3 and are described as follows:

- 1) Bits 1, 3, 4, 12, 13 and 14 are not used. They contain no data. These bits are always read as a DATA 0.
- 2) Bit 0 (Parity Error) -- If this bit is set (DATA1), the parity controller will assert BDAL17L during the data portion of a DATI or DATIO bus cycle. If this bit is not set (DATA0), the CPU will not trap to 114 on parity errors. The state of bit 0 can be changed by the processor by a DATO data transfer to the CSR. LSI-11 signal BINIT clears this bit.
- 3) Bit 2 (Write-Wrong Parity) -- If this bit is set (DATA1), the parity controller will assert BDAL16L to force the MSV11-E to generate even (incorrect) parity based on data transferred into memory (DATO/DATOB). A parity error is then caused on a read (DATI) cycle of this data. The state of bit 2 can be changed by the processor by DATO data transfer to the CSR. LSI-11 bus signal BINIT clears this bit.
- 4) Bits 5-11 (Error Address) -- Once a parity error has occurred, these bits contain the seven highest order address bits (A17-A11) of the faulty data which caused the parity error. The stored address bits describe the location of faulty data to within 1K of memory. Therefore, the memory module involved in the parity error can be located by using the partial address contained in bits 5-11. The software operating system has the ability to lock out (prevent the access of) a 1K block of memory that is specified by a programmer.

- 5) Bit 15 (Parity Error Bit) -- This bit is set (DATA1) by the parity controller when a parity error occurs. Bit 15 is a flag, but it does not cause a parity error trap in the processor. LSI-11 bus signal BINIT clears this bit.

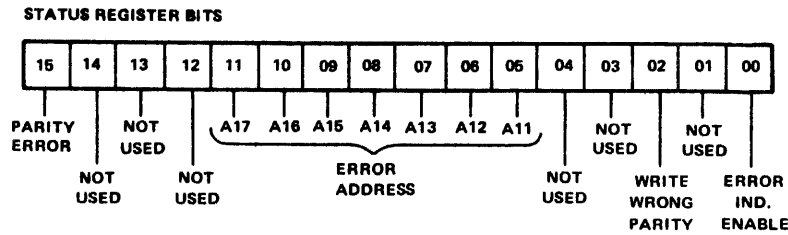


FIGURE 3 -- CONTROL AND STATUS REGISTER BIT ALLOCATION

The address of the CSR is determined in the parity controller circuitry; the address bits are shown in Figure 4. The address of the CSR is strapped within the range 772100 - 772136.

Address bit A00 is not decoded by the address select logic. Address bits (A04-A01) are determined by wired jumpers.

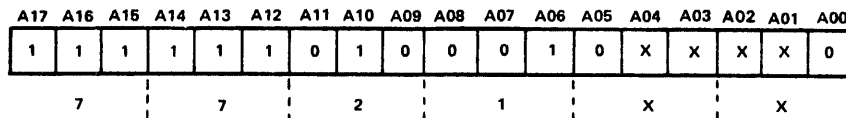


FIGURE 4 -- CSR ADDRESS BITS

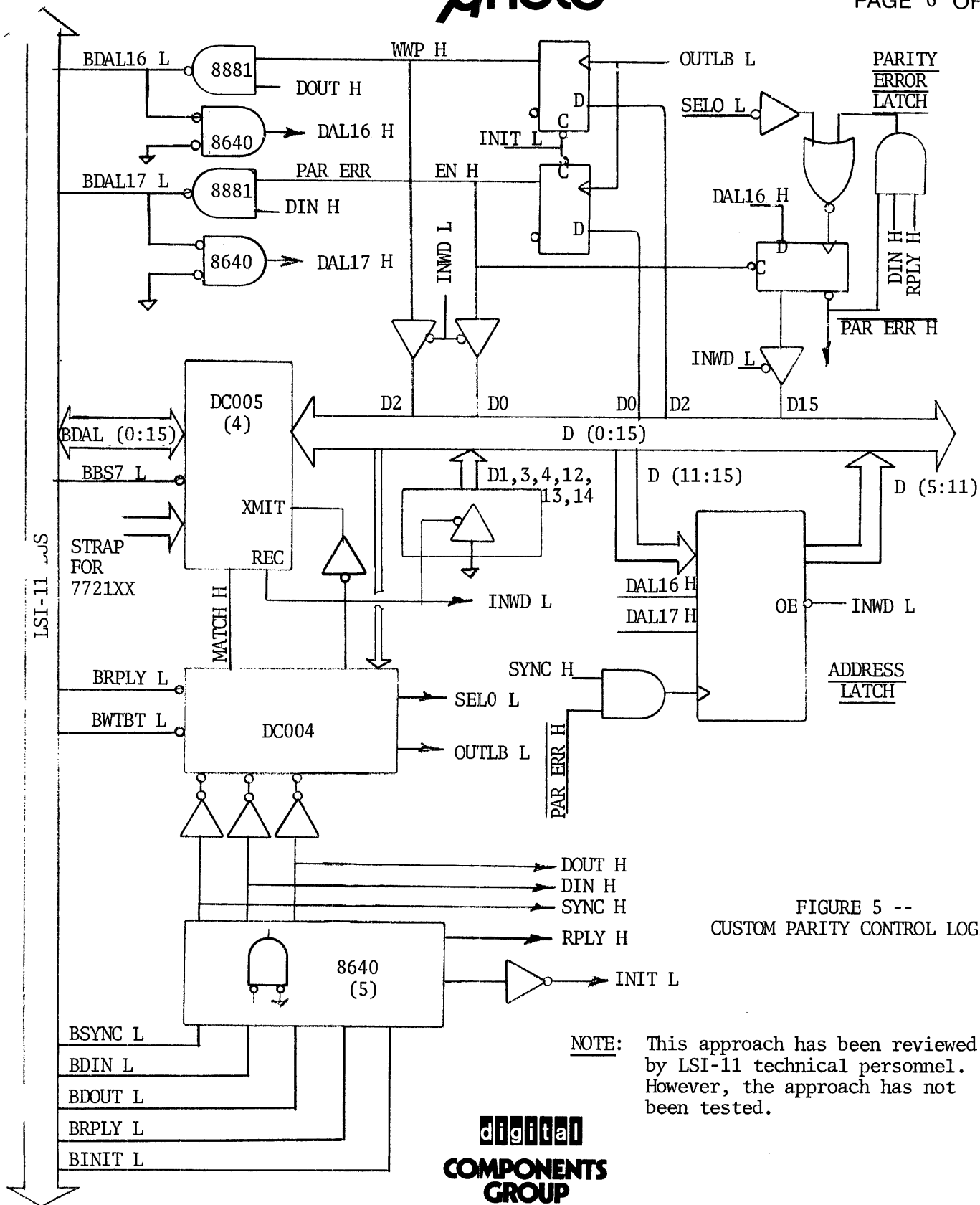


FIGURE 5 --  
CUSTOM PARITY CONTROL LOGIC

NOTE: This approach has been reviewed by LSI-11 technical personnel. However, the approach has not been tested.

CUSTOM PARITY CONTROL LOGIC

Figure 5 is a diagram of the logic required for the custom parity control module. This diagram is not intended to be the actual schematic for the parity controller but does show an approach that may be used. This approach has been thoroughly reviewed by LSI-11 technical personnel; however, the approach has not been tested.

The diagram breaks down into three distinct portions: the LSI-11 bus interface on the left, the parity error latch in the upper right-hand quadrant, and the address latch in the middle of the right-hand side. There is no block on the diagram that is called the Control and Status Register. In effect, the CSR is all of the logic on the diagram except for the bus interface logic.

The bus interface is fabricated out of integrated circuits available from DIGITAL. Except for some miscellaneous drivers and receivers (8881's and 8460's, respectively), all of the interface chips are available as part of the DCK11 Chipkit.

Only five of the six chips in the Chipkit are used. The DC003 interrupt protocol chip is not necessary for this application. Details on the bus interface can be found in Chipkit documentation. Outputs from the Chipkit interface include control signals and the Tri-State data bus D(0:15).

The parity error latch consists of a flip-flop and a pair of gates used to control the clock to the flip-flop. The flip-flop is set whenever the memory detects a parity error during a DATI bus cycle. If BDAL16 is asserted when the reply signal is asserted, the latch is set. The latch is cleared the first time the Control and Status Register is read by the processor. This flip-flop is bit 15 of the CSR.

The address latch is shown as a single block on this diagram. In reality, it is 7 flip-flops with Tri-State buffers on their outputs leading back to the data bus (this function may be performed by MSI chips similar to 74173's). On the leading edge of each BSYNC, the latch is clocked to save address bits 11-17. The clock is inhibited when a parity error occurs so that the upper address bits of the memory location where the parity error occurred are saved. The outputs of the address latch are CSR bits 5-11. These 7 bits may be used by the parity error service routine to determine which 1K block of memory the error occurred in.


The 2 flip-flops shown in the upper center of the diagram are used to implement CSR bits 0 and 2 (parity error enable and write-wrong parity, respectively). The lower flip-flop of the two is the parity error enable flip-flop and is set whenever the processor writes a 1 into bit 0 of the CSR. When parity error enable is set, BDAL17 is asserted whenever BDIN is asserted. This enables the processor to respond to parity errors. The write-wrong parity flip-flop (the upper flip-flop of the two) is set by writing a 1 into bit 2 of the CSR. When the write-wrong parity flip-flop is set, BDAL16 is asserted during the DOUT portion of the bus cycle. This forces the parity memory to store the wrong parity for diagnostic purposes.

Only one custom parity control module is necessary for all of the memory addressable by the KDF11-A processor.

#### SOFTWARE SUPPORT

When BDAL16 and BDAL17 are asserted during the data portion of a DATI cycle, the processor will recognize this as a parity error and trap to address location 114. The programmer must have provisions for responding to this trap. The service routine must read the Control and Status Register to find the address of that portion of memory where the parity error occurred and also to re-enable the parity error circuitry. The programmer must decide what course of action is to be followed when a parity error occurs.

RT-11 and RSX-11M have provision to support parity error traps.

	NUMBER 053
	DATE 12 / 27 / 78
	PRODUCT KD11, KDF11-A
	PAGE 1 OF 11
TITLE <u>PDP-11 Family Differences</u>	
DISTRIBUTION <u>Unrestricted</u>	
ORIGINATOR <u>Ted Semple</u>	

The attached PDP-11 Family Differences table will help users migrate their software between different members of the PDP-11 Family. Each member of the Family has some slight differences in the manner instructions are executed. Any program developed using PDP-11 operating systems with higher level languages will migrate with very little difficulty. However, some applications written in assembly language may have to be modified slightly. This table will help users determine where potential problems may occur.









ACTIVITY	LSI-		PDP-11					
	11	11/23	04	05/10	15/20	34	35/40	45
13. "T" bit trap will sequence out of WAIT instruction.		X	X	X	X	X	X	
----- "T" bit trap will not sequence out of WAIT instruction. Waits until an interrupt.	X							X
14. Explicit reference (direct access) to PS can load "T" bit. Console can also load "T" bit.			X	X	X			
----- Only implicit references (RTI, RTT, traps and interrupts) can load "T" bit. Console cannot load "T" bit.	X	X				X	X	X
15. Odd address/non-existent references using the SP cause a HALT. This is a case of double bus error with the second error occurring in the trap servicing the first error. Odd address trap not in LSI-11 or LSI-11/23.	X		X	X	X	X		
----- Odd address/non-existent references using the stack pointer cause a fatal trap. On bus error in trap service, new stack created at 0/2.		X					X	X
16. The first instruction in an interrupt routine will not be executed if another interrupt occurs at a higher priority level than assumed by the first interrupt.	X	X	X	X		X	X	X
----- The first instruction in an interrupt service is guaranteed to be executed.					X			
17. 8 general purpose registers.	X	X	X	X	X	X	X	
----- 16 general purpose registers								X
18. PSW address, 177776, not implemented must use new instructions, MTPS (move to PS) and MFPS (move from PS).	X							
-----								

ACTIVITY	LSI-		PDP-11					
	11	11/23	04	05/10	15/20	34	35/40	45
PSW address implemented, MTPS and MFPS not implemented.			X	X	X		X	X
PSW address and MTPS and MFPS implemented.		X				X		
19. Only one interrupt level (BR4) exists.	X							
Four interrupt levels exist.		X	X	X	X	X	X	X
20. Stack overflow not implemented.	X							
Stack overflow below 400 implemented.		X	X	X	X	X		
Red and yellow zone stack overflow implemented.							X	X
21. Odd address trap not implemented.	X	X						
Odd address trap implemented.			X	X	X	X	X	X
22. FMUL and FDIV instructions implicitly use R6 (one push and pop); hence, R6 must be set up correctly.	X							
FMUL and FDIV instructions do not implicitly use R6.							X	
23. Due to their execution time, EIS instructions can abort because of a device interrupt.	X							
EIS instructions do not abort because of a device interrupt.		X					X	X
24. Due to their execution time, FIS instructions can abort because of a device interrupt.	X						X	

ACTIVITY	LSI-		PDP-11					
	11	11/23	04	05/10	15/20	34	35/40	45
25. EIS instructions do a DATIP and DATO bus sequence when fetching source operand.	X							
----- EIS instructions do a DATI bus sequence when fetching source operand.		X					X	X
26. MOV instruction does just a DATO bus sequence for the last memory cycle.	X	X				X	X	X
----- MOV instruction does a DATIP and DATO bus sequence for the last memory cycle.			X	X	X			
27. If PC contains non-existent memory address and a bus error occurs, PC will have been incremented.	X	X	X	X	X	X		X
----- If PC contains non-existent memory address and bus error occurs, PC will be unchanged.							X	
28. If register contains non-existent memory address in mode 2 and a bus error occurs, register will be incremented.	X	X		X	X		X	X
----- Same as above but register is unchanged.			X			X		
29. If register contains an odd value in mode 2 and a bus error occurs, register will be incremented.	X	X					X	X
----- If register contains an odd value in mode 2 and a bus error occurs, register will be unchanged.			X	X	X	X		
30. Condition codes restored to original values after FIS interrupt abort (EIS doesn't abort on 35/40).							X	
----- Condition codes that are restored after EIS/FIS interrupt abort are indeterminate.	X							

ACTIVITY	LSI-		PDP-11					
	11	11/23	04	05/10	15/20	34	35/40	45
31. Op codes 075040 thru 075377 unconditionally trap to 10 as reserved op codes. ----- If KEV11 option is present, op codes 075040 thru 075377 perform a memory read using the register specified by the low order 3 bits as a pointer. If the register contents are a non-existent address, a trap to 4 occurs. If the register contents are an existent address, a trap to 10 occurs if user microcode is not present. If no KEV11 option is present, a trap to 10 occurs.		X	X	X	X	X	X	X
32. Op codes 210 thru 217 trap to 10 as reserved op codes. ----- Op codes 210 thru 217 are used as a maintenance instruction.		X	X	X	X	X	X	X
33. Op codes 075040 thru 075777 trap to 10 as reserved op codes. ----- Only if KEV11 option is present, op codes 075040 thru 075377 can be used as escapes to user microcode. Op codes 075400 thru 075777 can also be used.  As escapes to user microcode and KEV11 option need not be present. If no user microcode exists, a trap to 10 occurs.		X	X	X	X	X	X	X
34. Op codes 170000 thru 177777 trap to 10 as reserved instructions. ----- Op codes 170000 thru 177777 are implemented as floating point instructions. ----- Op codes 170000 thru 177777 can be used as escapes to user microcode. If no user microcode exists, a trap to 10 occurs.			X	X	X		X	
		X				X		
	X							

ACTIVITY	LSI-		PDP-11					
	11	11/23	04	05/10	15/20	34	35/40	45
35. CLR, SXT, MFPS, MTPI and MTPD do just a DATO sequence for the last bus cycle.		X						
----- CLR and SXT do DATIP-DATO sequence for the last bus cycle.	X		X	X	X	X	X	X
36. MEM.MGT maintenance mode SR0 bit 8 is implemented.						X	X	X
----- MEM.MGT maintenance mode SR0 bit 8 is not implemented.		X						
37. PS (15:12), user mode, user stack pointer, and MTPX and MFPX instructions exist even when MEM.MGT is not configured.		X						X
----- PS (15:12), user mode, user stack pointer and MTPX and MFPX instructions exist only when MEM.MGT is configured.							X	
38. Current mode PS bits (15:14) set of 01 or 10 will cause a MEM.MGT trap upon any memory reference.						X	X	X
----- Current mode PS bits (15:14) set to 01 or 10 will be treated as user mode (11) and not cause a MEM.MGT trap.		X						
39. MTPS in user mode will cause MEM.MGT trap is PS address 177776 not mapped. If mapped PS (7:5) and (3:0) affected.						X		
----- MTPS in user mode will only affect PS (3:0) regardless of whether PS address 177776 is mapped.		X						
40. MFPS in user mode will cause MEM.MGT trap is PS address 177776 not mapped. If mapped, PS (7:0) are accessed.						X		
-----								



ACTIVITY	LSI-		PDP-11					
	11	11/23	04	05/10	15/20	34	35/40	45
MFPS in user mode will access PS (7:0) regardless of whether PS address 177776 is mapped.		X						
41. A HALT instruction in user mode traps to 4.								X
----- A HALT instruction in user mode traps to 10.		X				X	X	
42. If an RTT sets the T bit and the next instruction is an RTI, which clears the T bit, the T bit trap will not be taken.						X	X	X
----- Same as above, but the T bit trap will be taken.	X	X	X					

PRIORITY OF TRAPS AND INTERRUPTS

<u>LSI-11</u>	<u>LSI-11/23</u>	<u>PDP-11/04</u>	<u>PDP-11/05,10</u>	<u>PDP-11/15,20</u>
BUSERR Trap	CTLERR Trap	BUSERR Trap	BUSERR Trap	BUSERR Trap
Memory Refresh	MMU Trap	Trap Inst.	Trap Inst.	Trap Inst.
Trap Inst.	BUSERR Trap	Trace Trap	Trace Trap	Trace Trap
Trace Trap	PARERR Trap	STOVF Trap	STOVF Trap	STOVF Trap
PFAIL Trap	Trap Inst.	PFAIL Trap	PFAIL Trap	PFAIL Trap
Bus Halt Signal	Trace Trap	Device Interrupts	Device Interrupts	Console Halt
Event Line	STOVF Trap	Console Halt	Console Halt	Device Interrupts
Device Interrupts	PFAIL Trap	Wait Loop	Wait Loop	Wait Loop
Wait Loop	Device Interrupts			
	Bus Halt Signal			
	Wait Loop			
<u>PDP-11/34</u>	<u>PDP-11/35,40</u>	<u>PDP-11/45</u>		
ODDAD Trap	PARERR Trap	Console Halt		
MMU Trap	MMU Trap	ODDAD Trap		
BUSERR Trap	BUSERR Trap	STOVF Trap		
PARERR Trap	STOVF Trap	(Red Zone)		
Trap Inst.	(Red Zone)	MMU Trap		
Trace Trap	Trap Inst.	BUSERR Trap	BUSERR = Timeout Error	
STOVF Trap	Trace Trap	PARERR Trap	MMU = Memory Management Trap	
PFAIL Trap	STOVF Trap	STOVF Trap	PARERR = Parity Error	
Device Interrupts	(Yellow Zone)	(Yellow Zone)	ODDAD = Odd Address Error	
Console Halt	PFAIL Trap	PFAIL Trap	STOVF = Stack Overflow	
Wait Loop	Console Halt	PIRQ	PFAIL = Power Fail	
	Device Interrupts	Device Interrupts	Inst. = Instructions	
	Wait Loop	Wait Loop		
		Trace Trap		

<b>μnote</b>		NUMBER 054
TITLE <u>MXV11 Configuration</u>		DATE 12 / 27 / 78
DISTRIBUTION <u>Unrestricted</u>		PRODUCT MXV11
ORIGINATOR <u>Joe Austin</u>		PAGE 1 OF 5

The MXV11 multi-function module contains the following functional elements:

- a) RAM - 8KB or 32KB with internal refresh
- b) ROM - sockets for: 2KB, 4KB or 8KB program memory  
512B bootstrap memory
- c) 2 DLV11-J type serial line interface ports
- d) 60 Hz crystal clock

1. This module will initially be available with two configurations of memory as follows:

MXV11-AA -- 8KB of RAM (ROM not included)

MXV11-AC -- 32KB of RAM (ROM not included)

The RAM can be configured to start on any 8KB boundary below 64KB. Because of this restriction, the MXV11-AA 8KB version is not usable for memory above 56KB. It can be used in 18-bit memory address systems, but it is restricted to being assigned to the lower memory area below 56KB.

The MXV11-AC, on the other hand, can operate above 56KB, but its starting address must be at 56KB (160000<sub>8</sub>) or below. This then allows it to cover the 56KB-88KB range (160000<sub>8</sub> to 257777<sub>8</sub>) as shown in Figure 1.

Beyond these restrictions, the MXV11-AA or MXV11-AC can be used with any other valid combination of memories.

2. The ROM area of the MXV11-AA or -AC can be configured to operate in bank 0 (0-8KB), 1 (8KB-16KB), or the I/O page (see Figure 2). If bank 0 or 1 is selected, then the customer can use his own 1Kx8, 2Kx8 or 4Kx8 ROMs, and the ROMs can be either fusible link PROMs, ultra-violet erasible EPROMs, or masked ROMs (see Micro Note #51).

If the boot area in the I/O page is selected, then the customer can use two 255x8 ROMs starting at the power up address of 173000<sub>8</sub>. This ROM area covers the 512 byte address range of 173000 to 173777. In addition, a configuration jumper is provided which allows a user to install two 512x8 ROMs. 512 bytes of the 1KB can be used at a time. A jumper is provided on the board to select the upper or lower half of the larger ROM.

The user may provide his own ROM in this area, or he may use the following DEC-supplied bootstrap ROM:

```
MXV11-A2 - ROM bootstraps for:  disk -- RLV11 (RL01)
                                RKV11 (RK05)
                                RXV21 (RX02)
                                RXV11 (RX01)
                                tape -- TU58
```

This ROM is a 512 word ROM that uses the above-mentioned jumper to select either the disk boots or the TU58 boot. It is not possible for both sets of boots to be accessible in the same system simultaneously, even if two MXV11 modules are used since both ROMs would occupy the same address space.

3. The two serial line ports have the same functionality as the DLV11-J except that RS-422 is not supported. The output connectors are the same, allowing the use of the same cables. The DLV11-KA option is also required to support a 20 mA interface.
  - a) Address Selection -- Serial port 0 may be assigned to one of four starting addresses: 176500, 176510, 176520 and 176530.  
  
Serial port 1 may be assigned addresses in two ranges. The first range starts at 176500 and covers the eight starting addresses from 176500 to 176570. The second range starts at 177500 and also contains eight possible starting addresses, including the standard console address, 177560. Due to the fact that several other standard DEC devices use addresses in this second range, it is recommended that only the console address be used.
  - b) Interrupt Vector Selection -- Vectors can be configured for either of two ranges -- 000 to 074 or 300 to 374. Both ports must be configured in the same range, but within the range any combination can be configured.
4. System Clock -- The 60 Hz crystal controlled clock can be jumpered to the BEVNT line to provide the equivalent of line time clock which does not otherwise have one. The factory configuration is to have this signal disconnected. It should not be connected if there is any other source in the system. This includes the case where there is more than one MXV11 module in a system.

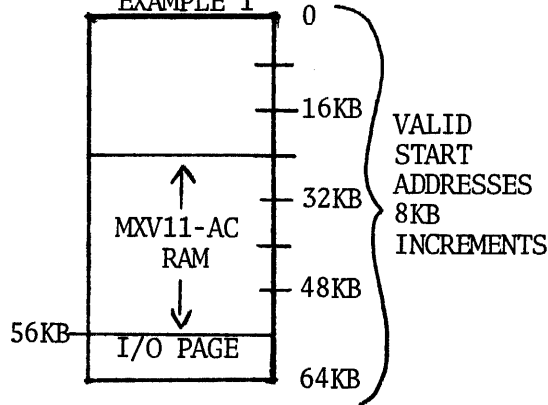
This clock can be used in conjunction with the BDV11 clock status/control register feature. The BDV11 can still be used to turn the clock off under program control since it accomplishes this by pulling the BEVNT L line to ground on the bus. If this control feature is to be used, the MXV11 should be installed in the same expansion box as the BDV11.

5. System Concepts:

- a) The ROM and RAM memories should not be configured to cover the same area of memory. There is no overlay protection logic to prevent conflicts in this case.
- b) The RAM memory will not respond to addresses in the I/O page area (bank 7 in 16-bit address systems). This prevents conflicts when peripherals (including the on-board SLUs) are addressed.
- c) It is not practical to install more than 2 MXV11 modules into a single system due to cost considerations. Even though the hardware limitation imposed by the SLU addressing is 4 modules, it is not recommended that customers consider using more than 2 in one system.

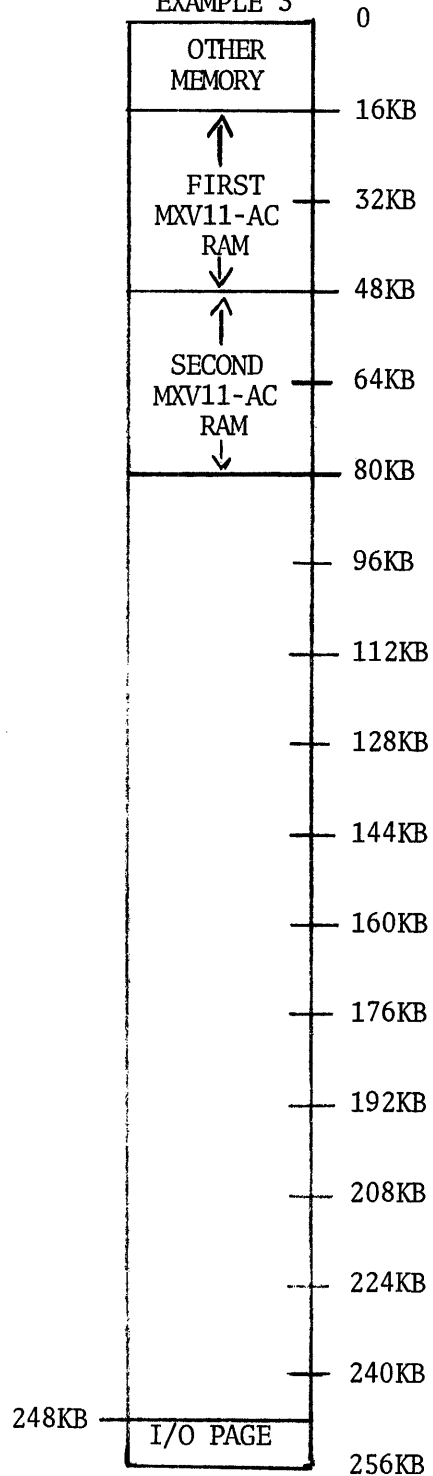
SMALL SYSTEM

SINGLE MXV11-AC  
EXAMPLE 1



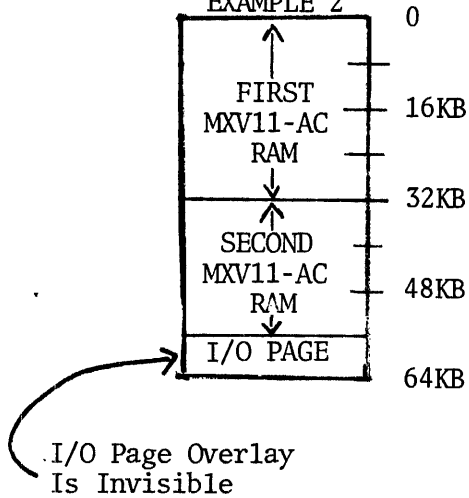
LARGE SYSTEM

DUAL MXV11-AC  
EXAMPLE 3



SMALL SYSTEM

DUAL MXV11-AC  
EXAMPLE 2



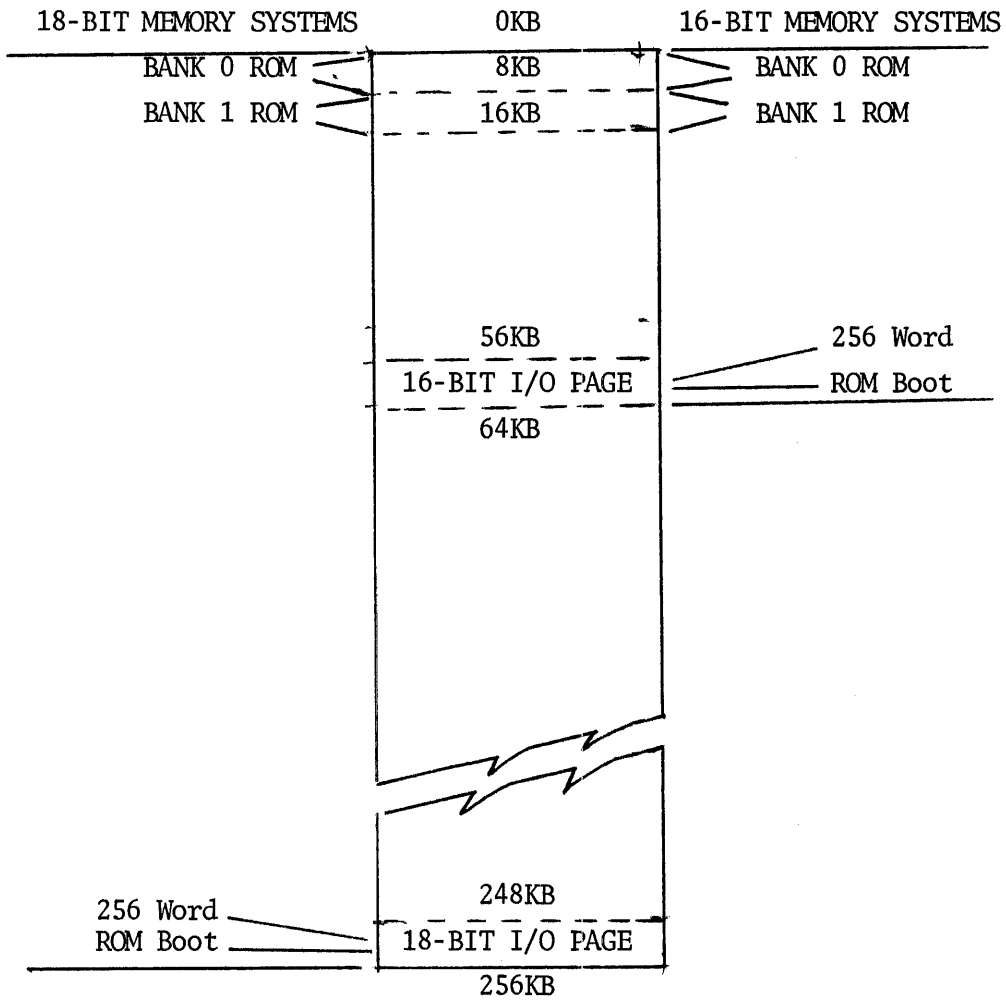



FIGURE 2 -- MXV11-AA, -AC ROM ADDRESSING RANGES

	NUMBER 055
	DATE 12 / 28 / 78
	PRODUCT KD11; KDF11-A
	PAGE 1 OF 2
TITLE LSI-11 vs. LSI-11/23 Bus Timing	
DISTRIBUTION Unrestricted	
ORIGINATOR Ted Semple & Doug Swartz	

Even though the LSI-11 bus is asynchronous, many users did not take advantage of this feature by optimizing their custom interfaces. Undoubtedly, some custom interfaces that work with LSI-11 and LSI-11/2 processors will not work with the LSI-11/23 because the KDF11-A module performs bus transactions faster than the KD11 modules. This Micro Note will help users isolate problems that might arise when they migrate their applications to the new processor.

NOTE: Any custom interface that meets the LSI-11 bus specification published in the LSI-11 Processor Handbook will work with either the KD11 or KDF11-A processor modules. Only modules that do not abide by the specification have a potential compatibility problem.

Note the following while studying Table 1 - LSI-11 vs. LSI-11/23 Timing Differences:

- 1) The KDF11-A performs bus transactions faster than the KD11-F.
- 2) Neither the KDF11-A nor the KD11-F run at the maximum bus speed.

Users should review their custom interface designs to make sure that changing things like address set-up time from 285 ns. to 196 ns. will affect their module operation.

All DIGITAL modules meet the bus timing specification and will work with all processor modules. (Some modules such as the REV11 and RKV11 are not compatible with the KDF11-A processor, but this is not because of bus timing.) Likewise, modules designed with the DCK11 Chipkits will work properly with all processors.




TABLE 1LSI-11 vs. LSI-11/23 BUS TIMING DIFFERENCES

	<u>BUS SPEC</u> <sup>1</sup>	<u>KD11-F</u> <sup>2</sup>	<u>KDF11-A</u> <sup>3</sup>
BSYNC L - BDIN L	100 ns. min.	190 ns.	130 ns.
BSYNC L - BDOUT L	200 ns. min.	285 ns.	260 ns.
ADDRESS SET-UP TIME	150 ns. min.	285 ns.	196 ns.
ADDRESS HOLD TIME	100 ns. min.	100 ns.	100 ns.
REPLY TO DIN/DOUT INACTIVE TIME	150 ns. min.	<u>720 ns. min.</u> 1120 ns. max.	<u>220 ns. min.</u> 285 ns. max.

NOTES:

1. Bus specification times are at the bus master inside the bus drivers.
2. KD11-F with 380 ns. microcycle time.
3. KDF11-A with 300 ns. microcycle time.

	NUMBER 056
	DATE 12 / 29 / 78
	PRODUCT DLV11-J
	PAGE 1 OF 4
TITLE <u>DLV11-J Cabling</u>	
DISTRIBUTION <u>Unrestricted</u>	
ORIGINATOR <u>Barbara Beck</u>	

This Micro Note presents all the cables currently available that will mate with the 2x5 pin Amp connector on the DLV11-J, as well as some pointers and part numbers for constructing a cable.

DEC cables for the DLV11-J:

- |          |   |
|----------|---|
| BC20N-05 | 5' EIA RS-232C null modem cable to directly interface with an EIA RS-232C terminal (2x5 pin Amp female to RS-232C female; see Figure 3).  |
| BC21B-05 | 5' EIA RS-232C modem cable to interface with modems and acoustic couplers (2x5 pin Amp female to RS-232C male; see Figure 2).   |
| BC20M-50 | 50' EIA RS-422 or RS-423 cable for high-speed transmission ( 19.2K baud) between two DLV11-J's (2x5 pin Amp female to 2x5 pin Amp female).  |
| DLV11-KA | 20 mA current loop converter option for the DLV11-J. Comes with an EIA cable (BC21A-03) which connects the DLV11-KB converter box to the DLV11-J. The option mates with standard DEC 20 mA cabling using the 8-pin mate-'n'-lock connector. |

When designing a cable for the DLV11-J, here are several points to consider:

1. The receivers on the DLV11-J have differential inputs. Therefore, when designing an RS-232C or RS-423 cable, RECEIVE DATA (pin 7 on the 2x5 pin Amp connector) must be tied to signal ground (pins 2, 5, or 9) in order to maintain proper EIA levels. RS-422 is balanced and uses both RECEIVE DATA+ and RECEIVE DATA-.
2. To directly connect to a local EIA RS-232C terminal, it is necessary to use a null modem. To design the null modem into the cable, one must switch RECEIVED DATA (pin 2) with TRANSMITTED DATA (pin 3) on the RS-232C male connector as shown in Figure 3.

3. To mate to the 2x5 pin connector block, the following parts are needed:

Cable Receptable	AMP PN 87133-5 DEC PN 12-14268-02
Locking Clip Contacts	AMP PN 87124-1 DEC PN 12-14267-00
Key Pin (pin 6)	AMP PN 87179-1 DEC PN 12-15418-00

4. The pin out on the 2x5 pin connector block on the DLV11-J is as follows:

<u>Pin #</u>	<u>Signal</u>
1	UART clock in or out (16 x baud rate; CMOS)
2	Signal ground
3	TRANSMIT DATA+
4	TRANSMIT DATA-
	Note: For EIA RS-423, this line is grounded. For DLV11-KA 20 mA option, this line is the reader enable pulse.
5	Signal ground
6	Index in key - no pin
7	RECEIVE DATA-
8	RECEIVE DATA+
9	Signal ground
10	When F1 is installed for the DLV11-KA, +12V is supplied through a 1A fuse to this pin

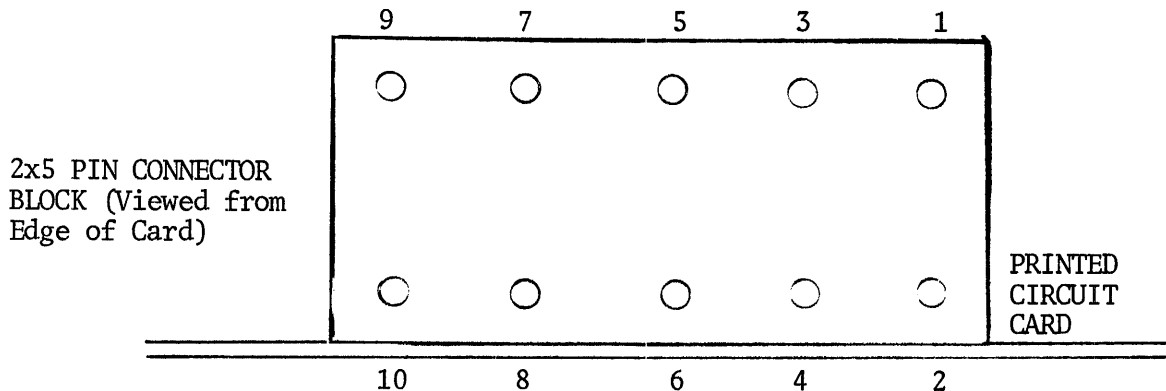
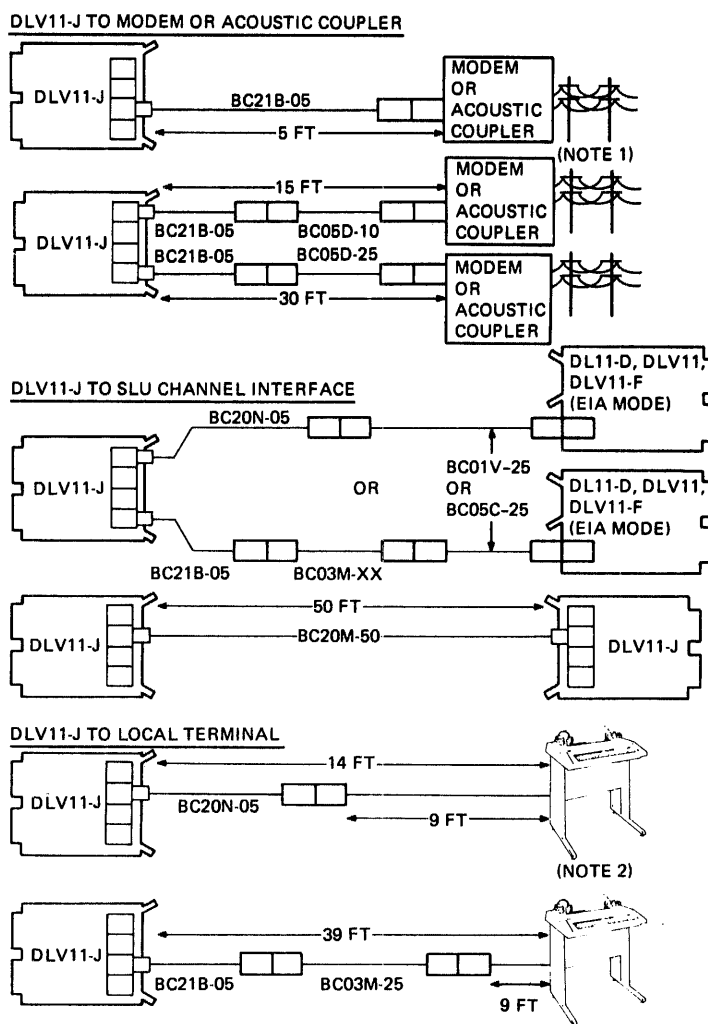
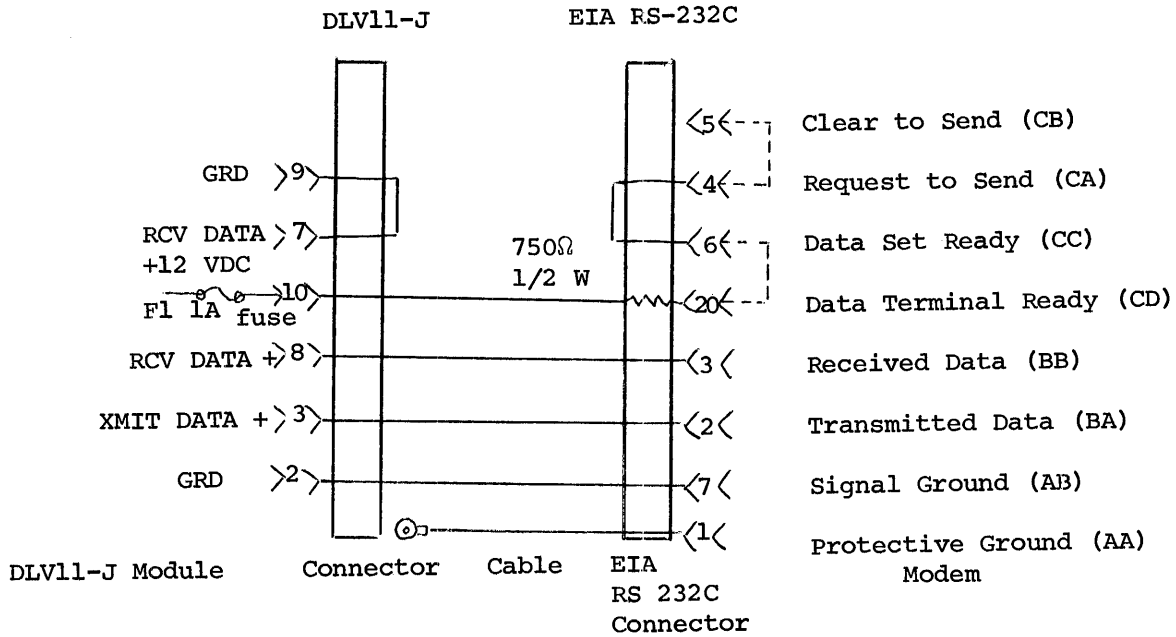


FIGURE 1  
DLV11-J CABLING SUMMARY



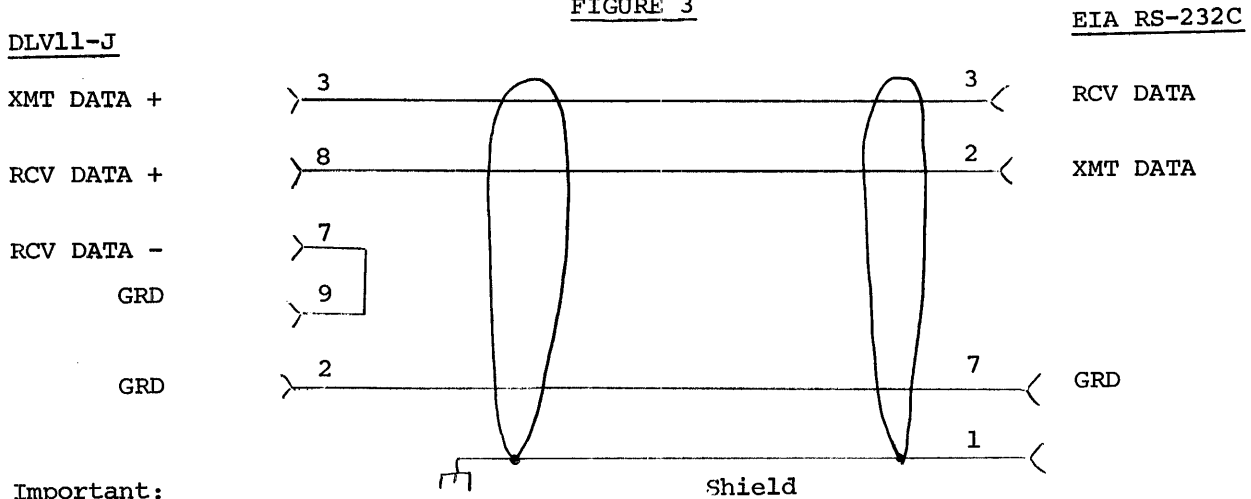
BC21B-05 MODEM CABLE

FIGURE 2




BC20N-05 'Null Modem' Cable

FIGURE 3



Important:  
Attach to chassis  
at entry point.

	NUMBER 057
	DATE 12 / 29 / 78
	PRODUCT BDV11-AA
	PAGE 1 OF 2
TITLE Location of W13 on the BDV11	
DISTRIBUTION Unrestricted	
ORIGINATOR Joe Austin	


W13 on the BDV11-AA module is referenced in several places in the Memories and Peripherals Handbook, but its location is not shown in any figures. The purpose of this Micro Note is to show where W13 is located.

W13 on the BDV11-AA module is located as follows (see Figure):

1. Remove capacitor C30 which is located between PROM sockets E37 and E38.
2. Insert one end of W13 in the pad that was used for the right-hand side of C30 as shown in the Figure.
3. Insert the other end of W13 in the feed-through hold located approximately 7/16 inch to the right of the pad of item 2 above.

The function of W13 is to connect either +5V or GND to pin 21 of same ROM sockets to allow for use by different types of ROMs.



 TITLE <u>Configuring Memory for LSI-11 Systems With More Than 64K Bytes</u>  DISTRIBUTION <u>Unrestricted</u>  ORIGINATOR <u>John Hughes</u>	NUMBER 058
	DATE 1 / 02 / 79
	PRODUCT LSI-11/23
	PAGE 1 OF 6

The LSI-11/23 makes it possible to implement LSI-11 Family systems with up to 248KB of memory (the last 8KB of address space is reserved for peripheral device addresses). The LSI-11 and LSI-11/2 processors are capable of addressing up to 56KB of memory, with a total of 64KB of address space including I/O addresses. There are a number of characteristics of the memory modules in the LSI-11 Family which will affect their usefulness in large memory (more than 64KB) configurations. This Micro Note focuses on the considerations involved in configuring large memory LSI-11 systems.

#### MEMORY CHARACTERISTIC DESCRIPTION

Attached to this Micro Note is a table describing four different characteristics for each of the memory modules in the LSI-11 Family. The following points summarize the memory characteristics and their impact on large memory configurations.

##### Address Lines Decoded

16 address lines are required to decode addresses up to 64KB. Some of the earlier LSI-11 Family modules decoded only 16 lines because this was the maximum address capability of the LSI-11 and LSI-11/2. These modules are not useful in large memory configurations. Only which modules which decode 18 address lines can be used for larger than 64KB configurations.

##### Start Address Increments

The start addresses for memory modules are selected by straps or switches on the memory board. The start address increments entry in the attached table indicate the granularity of start addresses that can be selected. An 8KB entry in this column indicates that start addresses 0, 8K, 16K, 24K, etc. are possible.

Although this memory characteristic does not impact the usefulness of memory modules in large memory configurations, it does impact the way in which memories are positioned in address space.



## Start Address Positioning

Start address ranges for memory modules fall into three categories:

- 1) Start addresses that are in the first 64K bytes of address space.
- 2) Start addresses that can be anywhere within the 256KB address space of the LSI-11/23.
- 3) Start addresses that are at the standard bootstrap location (173000 for a small memory system and 773000 for a large memory system).

All memory modules which have 16-bit address capability are confined to start in the first 64KB of address space. Most, but not all, memories which have 18-bit addresses can start anywhere within the 256KB address space of the LSI-11/23. The exception to this rule is the MXV11. This module, although it has 18-bit addressing, has a RAM memory section which is confined to start addresses in the first 64KB.

## Capacity

The column headed capacity shows the amount of storage provided by each memory module in K bytes. Some memory modules have varying capacity. Capacity variations are accomplished by using memory chips of varying densities and also by partially or completely populating memory boards with memory chips.

## Miscellaneous Memory Characteristics

- 1) RAM/PROM Implementation -- RAM memory boards are provided with the memory chips permanently installed (soldered) on the board. PROM memory boards, on the other hand, are provided with PROM sockets only. Customers are advised to purchase one of the chip types which are recommended for each of the PROM boards.
- 2) MSV11-B Refresh -- The MSV11-B memory is the only memory which requires bus refresh. This refresh has to be provided either by one of the quad size LSI-11 processors using microcode refresh or alternately, by DMA using an REV11. All other RAM memory modules are self-refreshed; that is, they have the circuitry on the memory board to refresh the dynamic RAM chips independent of any bus activity. The MSV11-B is a relatively low density memory board. It is not recommended for new designs.

### 3) MXV11 Configurations

The MXV11 multi-function board warrants separate consideration because it is a very popular board with some special configuration characteristics. The following points summarize those characteristics:

- 1) RAM memory capacity for the MXV11 can either be 8KB or 32KB.
- 2) RAM memory start addresses must be in the range of 0 to 56KB in increments of 8KB. The memory address space that is covered by the MXV11 can begin in the first 64KB addresses and extend into the next 64KB addresses. The start address cannot, however, be outside the range of the first 64KB.
- 3) There are three alternatives for the start address for PROM memory on the MXV11:
  - a) address 0
  - b) address 20,000<sub>8</sub>
  - c) address 173000<sub>8</sub> (bootstrap)

When the 32KB RAM version of the MXV11 is purchased, a maximum of two of these modules can be used in an LSI-11/23 system.

### 4) Peripheral Address Decoding

8KB of addresses is reserved in all LSI-11 Family systems for addressing peripheral devices. In small memory systems with a maximum of 64KB addresses, the peripheral device address space is in the address range of 56KB to 64KB. In a large memory system with a total of 256KB addresses, the peripheral device address space is in the range of 248KB to 256KB.

Peripheral devices decode the low order 13 bits of an address and a separate line called BBS7 to determine when they are being addressed. They do not decode the high order address bits. BBS7 is generated by the processor any time that it wishes to communicate with a peripheral device in the upper 8KB of address space. This type of arrangement means that it is unnecessary to change peripheral devices to operate in small (64KB) or large memory (256KB) configurations.

When configuring a large memory system, it is possible to position memory in the range of 56KB to 64KB. This space would normally be reserved for peripheral device addresses in a small memory configuration.

5) MRV11-C Window Mapped

The MRV11-C has two modes of operating. It can either be mapped directly into the address space of the LSI-11 bus or it can appear as two separate 1K address windows. The window concept makes it possible for small memory configurations to have access to a large amount of PROM storage without taking up a large amount of address space. A separate Micro Note in the future will describe the window mapping concept in more detail.

6) MSV11-D I/O Page Overlap

The MSV11-D has the ability to provide 60KB of memory in a small memory system with a maximum address space of 64KB. This is accomplished by mapping RAM memory into the address space between 56KB and 60KB which would normally be reserved for peripheral device addresses. This feature of the MSV11-D is available only on the MSV11-DD model where the start address is configured to be location 0.

This capability is also available for the MSV11-DD in large memory configurations provided that the start address is 192KB. With the I/O page overlap feature enabled in this large memory configuration, the MSV11-DD configuration will provide memory all the way up to 252KB overlapping half of the I/O address space (i.e., 4KB).

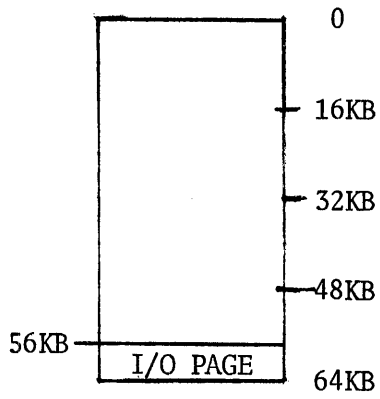
MEMORY MAPS

Included in this Micro Note is a pair of memory maps for small and large systems. These maps are intended to be used as a tool in configuring memory for LSI-11 systems.

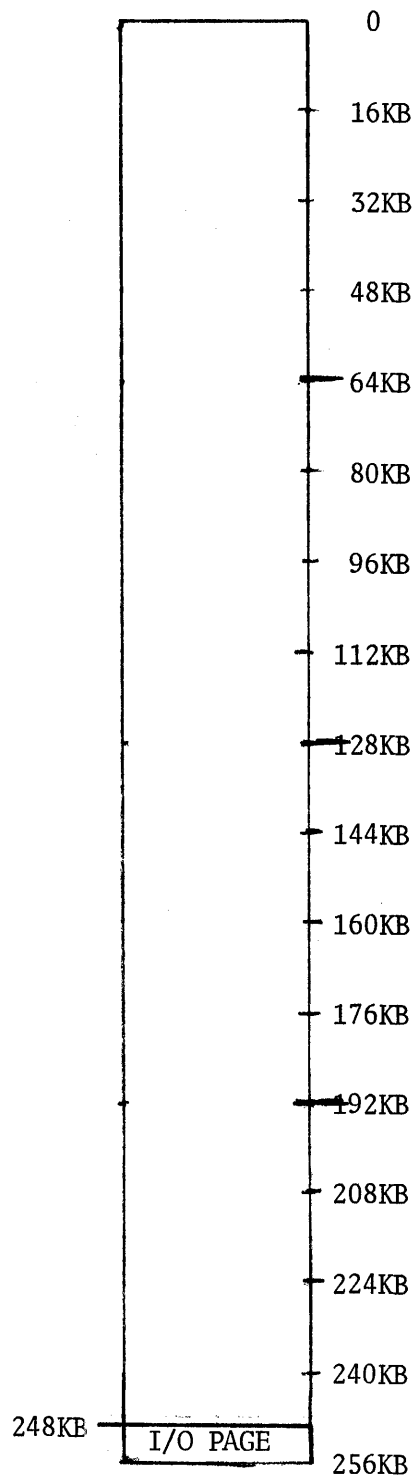
LSI-11 MEMORY CHARACTERISTICS


	<u>ADDRESS LINES DECODED</u>	<u>START ADDRESS INCREMENTS</u>	<u>START ADDRESS POSITIONING</u>	<u>CAPACITY</u>
<u>RAM</u>				
MMV11-A	16	8KB	0-56KB	8KB
MSV11-B	16	8KB	0-56KB	8KB
MSV11-CD	18	8KB	0-248KB	32KB
MSV11-D A	18	8KB	0-248KB	8KB
B	18	8KB	0-248KB	16KB
C	18	8KB	0-248KB	32KB
D	18	8KB	0-248KB	64KB
<u>PROM</u>				
MRV11-AA (512x4 PROMs)	16	8KB	0-56KB	1KB-8KB
(256x4 PROMs)	16	4KB	0-56KB	512 Bytes-4KB
MRV11-BA PROM	18	8KB	0-248KB	2KB-8KB
RAM	18	512 Bytes	0-248KB	512 Bytes
MRV11-C	18	8KB	0-248KB	8KB-64KB
<u>MULTI-FUNCTION</u>				
MXV11 RAM	18	8KB	0-56KB	8KB or 32KB
PROM	18	8KB	0-8KB (or boot)	2KB-8KB

SMALL SYSTEM  
MEMORY MAP



LARGE SYSTEM  
MEMORY MAP



	NUMBER 059
	DATE 1 / 04 / 79
	PRODUCT LSI-11/23
	PAGE 1 OF 7
TITLE LSI-11/23 Four-Level Interrupts	
DISTRIBUTION Unrestricted	
ORIGINATOR Dave Schanin	

## INTRODUCTION

The LSI-11/23 implements a four-level priority interrupt scheme which is backward compatible to the single level system used on the LSI-11 and LSI-11/2. This allows a prioritizing of the interrupts so that a high priority interrupt request can interrupt a lower priority service routine. Figure 1 illustrates the LSI-11 priority scheme. Figure 2 illustrates the LSI-11/23 priority scheme. Note that the LSI-11/23 has both a vertical system priority (that is software controllable) as well as the LSI-11 type of horizontal priority within each system priority level.

## FOUR-LEVEL INTERRUPT IMPLEMENTATION

### Hardware - Signal Line Definition

On the LSI-11, the BIRQ line was treated as a level four interrupt; i.e., BIRQ 4. The LSI-11/23 preserves this definition and redefines three additional lines to gain the additional three levels of interrupt:

BSPARE 6	BIRQ 7
BSPARE 2	BIRQ 6
BSPARE 1	BIRQ 5

### Hardware - Interrupt Acknowledge Scheme

The LSI-11 interrupt acknowledge scheme consisted of a daisy-chained grant signal issued from the processor whenever the BIRQ line was asserted. It was the responsibility of the I/O modules to accept the IAKI pulse and pass it on as an IAKO pulse if the module was not requesting an interrupt. IAKO would not be passed on if the module was awaiting an interrupt acknowledge. The LSI-11/23 preserves this IAK scheme for backward compatibility with the LSI-11 but adds an additional responsibility to the I/O modules to pass IAK if a device is requesting an interrupt but there is a higher priority interrupt request pending.

## Hardware - Interrupt Protocol

There are two methods for handling interrupt protocol. The first method uses position-dependent I/O modules and the second method uses position-independent modules within interrupt levels.

The primary motivation for the two different methods of implementing four-level interrupts has to do with whether the module under discussion already exists or is in design. The Position-Dependent approach allows simple and straightforward upgrade of an existing module interfaced to the LSI-11 bus to operate in a multi-level environment on any interrupt level other than four. The Position-Independent approach is recommended for new module implementations; it is the approach that will be used by DIGITAL on all future modules.

As a general description, position-dependent I/O modules must not only be in priority order within their respective levels but must also be in system priority order (see Figure 3). Position-independent modules, however, can be placed anywhere in the backplane and still maintain their assigned interrupt priority level. These modules need only be placed in priority order within their respective priority levels. The most significant drawback to the module position-dependent approach is non-backward compatibility with the LSI-11. This will be explored in the following two sections.

### METHOD ONE - POSITION-DEPENDENT MODULES

Method One is simply an extension of the present LSI-11 interrupt scheme. A module which is requesting an interrupt simply asserts the BIRQ line for that particular interrupt level. The IAK acknowledge signal is also handled identically with the LSI-11. When an IAKI signal is received, the module passes the signal to the IAKO only if the module is not requesting an interrupt. The limitation of this simple scheme is that the modules in the system must be placed in interrupt priority order not only within an interrupt level but also must be placed within the system interrupt levels; i.e., all level 7 modules must be closer to the processor than level 6 modules, etc. An important observation should be made concerning this approach to four-level interrupts. If a module is designed to implement an interrupt level above BIRQ 4 and it only asserts BIRQ 5, 6, or 7, the module will not be transferrable to an LSI-11 based system since the only BIRQ line available in the LSI-11 is BIRQ 4.

C03-1200  
DISK  
CONTRIBUTE

### METHOD TWO - POSITION-INDEPENDENT MODULES

Method Two allows the modules to be placed in any sequence within the system; however, the modules must still be placed in order within an interrupt level. The basic approach here is that a module asserts the BIRQ line corresponding to the interrupt level that it is on and, in addition, asserts the BIRQ lines of lower priority according to Chart 1. The scheme outlined in Chart 1 requires

a module to monitor a maximum of two other BIRQ lines and assert a maximum of three BIRQ lines. This is done to allow design of a DC103 Integrated Circuit that has just two additional pins over the standard DC003 chip. The DC103 will be incorporated in new I/O module designs from DIGITAL. Therefore, it is important to adhere to the scheme outlined in Chart 1 to maintain the approach that DIGITAL will be following in the future.

In addition, there are two side benefits from this assertion/monitoring scheme. One is that a module so designed will be backward compatible with the LSI-11 since regardless of the interrupt level of the module, it will always assert BIRQ 4. The second is that it gains position independence for modules on separate interrupt levels. This is accomplished by modifying the interrupt acknowledge sequence as follows: A module monitors the BIRQ line that is at a higher priority level than the one on which it is currently requesting an interrupt. When such a module requests an interrupt and receives an IAKI, it will pass on the IAKO despite requesting an interrupt if the higher level BIRQ line is asserted. The module will retain the IAKI only if it is both requesting an interrupt and the higher level BIRQ line is asserted. The penalty here is the increased functionality required on the module. However, as was previously mentioned, this method retains backware compatibility with the LSI-11. Note that any module in this position-independent system needs to only monitor BIRQ 5, 6 because of the BIRQ assertion scheme in Chart 1. This is done to reduce the number of line receivers required on an I/O module and also allows the module to be compatible with the DC103 integrated circuit that will be used by DIGITAL to create four-level interrupt capability on the standard I/O modules. Therefore, an important feature of this second approach to four-level implementation is that it will be compatible with future modules from DIGITAL.

<u>MODULE INTERRUPT LEVEL</u>	<u>CHART 1 ASSERTS</u>	<u>MONITORS</u>
Level 4	BIRQ 4	BIRQ 5 BIRQ 6
Level 5	BIRQ 4 BIRQ 5	BIRQ 6
Level 6	BIRQ 4 BIRQ 6	BIRQ 7
Level 7	BIRQ 4 BIRQ 6 BIRQ 7	(None)

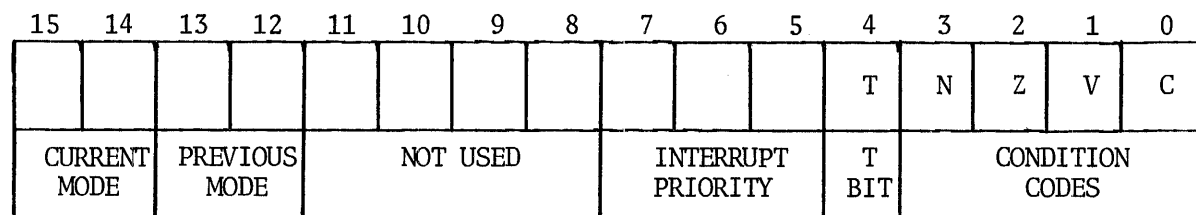
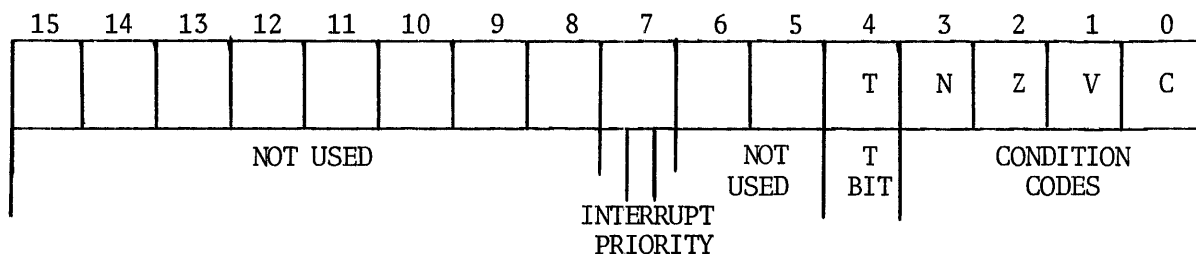


## HARDWARE - SUMMARY OF FEATURES

<u>APPROACH</u>	<u>POSITIVE FEATURES</u>	<u>NEGATIVE FEATURES</u>
Position Dependent Modules	<ol style="list-style-type: none"> <li>1. Easy to implement</li> <li>2. Can use DC003</li> </ol>	<ol style="list-style-type: none"> <li>1. Not compatible with LSI-11</li> <li>2. Not functionally the same as the approach to be used by DIGITAL</li> <li>3. Module placement in the backplane is critical</li> </ol>
Position Independent Modules	<ol style="list-style-type: none"> <li>1. Compatible with LSI-11</li> <li>2. Identical with the DIGITAL approach</li> <li>3. Module placement not critical to correct system operation</li> </ol>	<ol style="list-style-type: none"> <li>1. Requires added logic on the I/O board or DC103 chip</li> </ol>

## SOFTWARE IMPLICATIONS

The software has the capability of establishing the minimum priority level that is to be allowed to interrupt. This is done by manipulating the Processor Status Word (PSW). Bits 5-7 establish the minimum interrupt priority in an LSI-11/23 system. In the LSI-11 system, bit 7 (corresponding to 100<sub>2</sub> or 4 for BIRQ 4) was the only priority bit. Here is a comparison of the two PSW's:



It should be noted that all DEC software correctly handles either the single level or the multiple levels of interrupt priority.

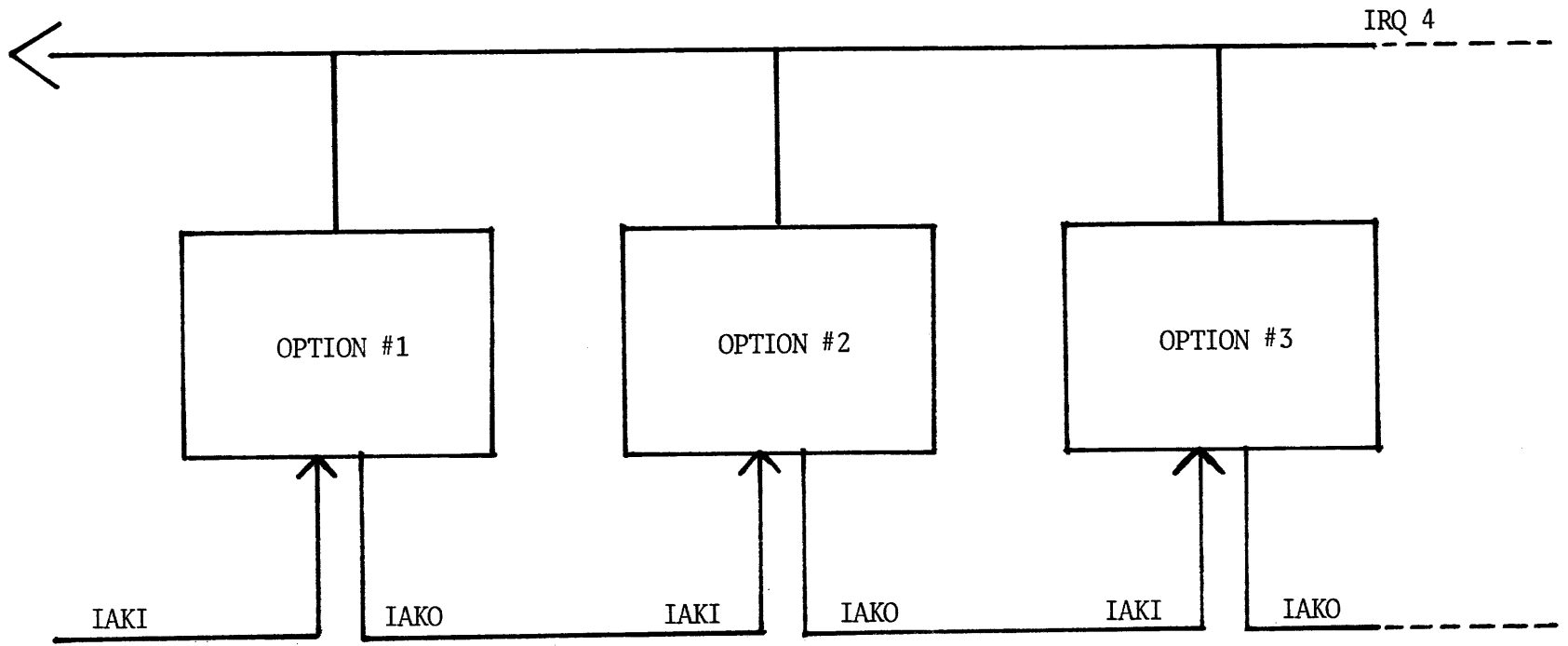


FIGURE 1 -- LSI-11 INTERRUPT SCHEME

FIGURE 2 -- LSI-11/23 POSI N-INDEPENDENT MODULES

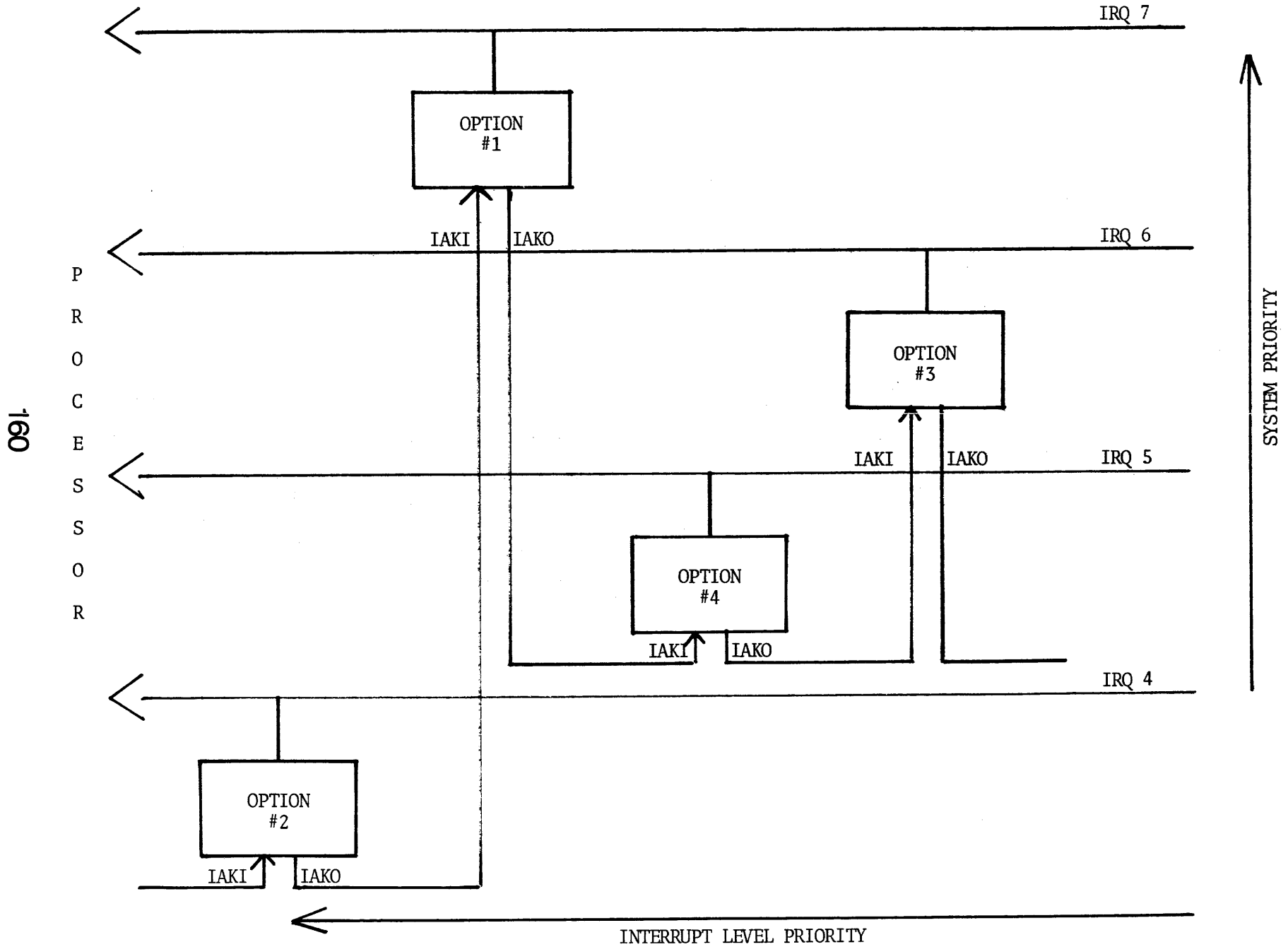
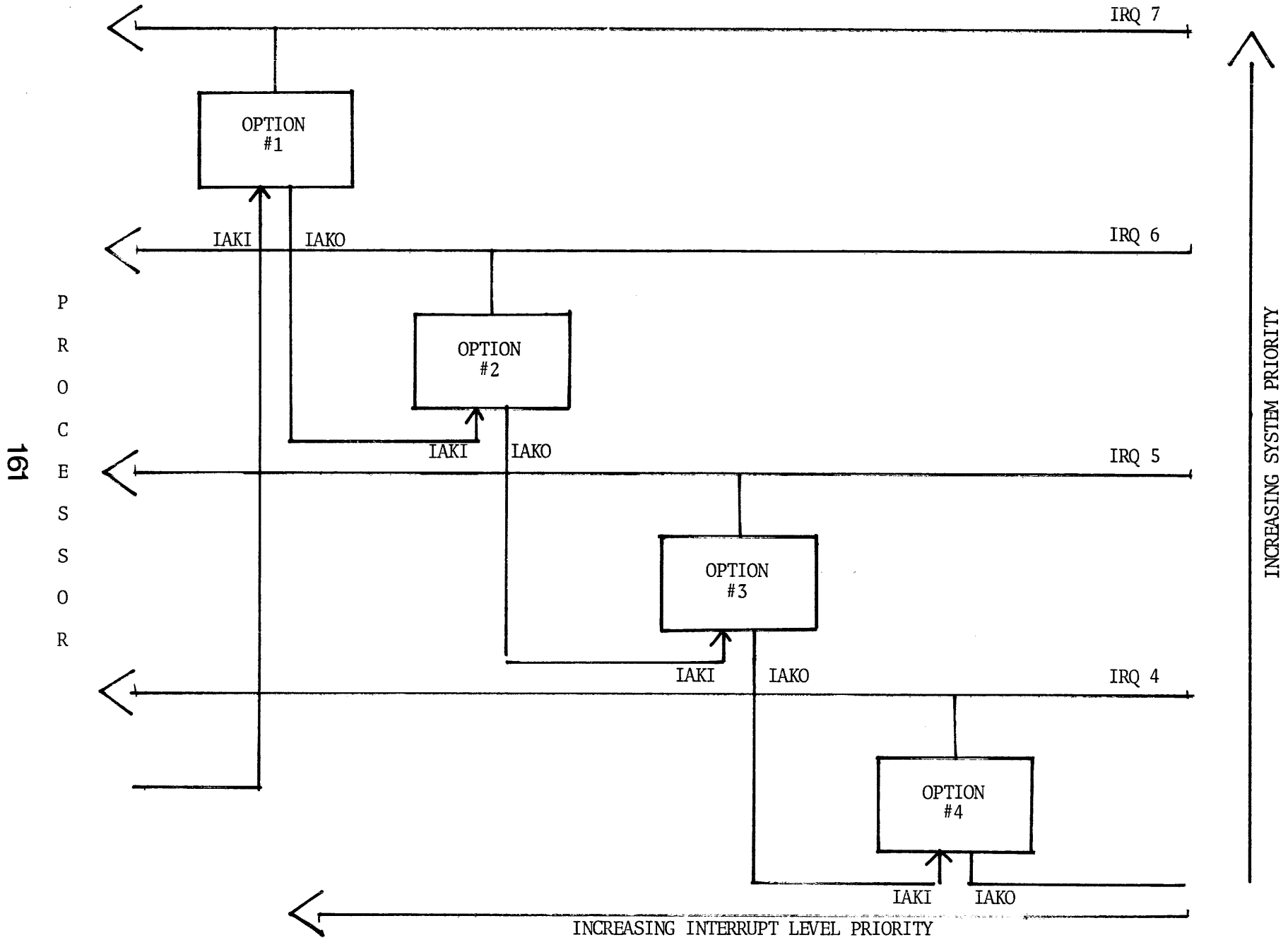



FIGURE 3 -- LSI-11/23 POE JN-DEPENDENT MODULES



 TITLE <u>Maximum Configuration of DLV11-J Modules</u> DISTRIBUTION <u>DLV11-J Users</u> ORIGINATOR <u>Joe Austin</u>	NUMBER 060
	DATE 3 / 01 / 79
	PRODUCT DLV11-J
	PAGE 1 OF 2

The purpose of this Micro Note is to define the maximum number of DLV11-J 4-line serial line unit modules that can be configured into a single system.

The maximum number of DLV11-J modules that can be installed into a single system is limited by the range of interrupt vectors that can be configured on the module. Starting with the base vector, the DLV11-J uses eight consecutive interrupt vectors. However, the DLV11-J module provides only three sets of configuration jumpers with which to configure the base vector.

The usable base vectors are shown below. Note that only the first two conform to DIGITAL standards and are free from conflicts with other options. Up to three more base vectors can be used if the conflicting options are not present in the system. In the case of modules #3 and #4, the vectors of the conflicting options can be reconfigured to another value. This would require changing both the hardware and the support software for the conflicting option. In the case of the 5th DLV11-J module, the conflicts with the FIX and MMU options cannot be resolved if either is present.

#### SUMMARY


- 2 DLV11-J modules (8 serial lines) can be easily configured in a single system.
- 5 DLV11-J modules (20 serial lines) can be configured if conflicts can be resolved with other options.

<u>DLV11-J</u>	<u>BASE VECTOR*</u>	<u>STANDARD ADDRESS</u>	<u>CONFLICTS WITH</u>	<u>REMARKS</u>
#1	300	176500		Std. DIGITAL Configuration
#2	340	176540		Std. DIGITAL Configuration
#3	140		RLV11	Non-Standard
#4	200		LPV11, RKV11	Non-Standard
#5	240		FIS, RXV11, MMU	Non-Standard

### ADDITIONAL SERIAL LINES

The DLV11, DLV11-E, DLV11-F and DZV11 can be used to add additional lines, if necessary, that do conform to DIGITAL standards. The assignment of addresses and vectors to conform to DIGITAL standards is defined in Appendix A of the Memories and Peripherals Handbook.

\*The base vector cannot be configured above 400.

 TITLE <u>Programming the MRV11-C</u> DISTRIBUTION <u>Unrestricted</u> ORIGINATOR <u>Rich Billig</u>	NUMBER 061
	DATE 2 / 02 / 79
	PRODUCT MRV11-C
	PAGE 1 OF 9

## INTRODUCTION

The MRV11-C is a high-density ROM memory module for the LSI-11 bus. The 16 24-pin sockets on this module will accept 1024x8, 2048x8, or 4096x8 ROM, PROM or EPROM memory chips. This gives the module a maximum storage capacity of 64K bytes. All chips used on a single board must be of the same density. However, the board may be partially populated in pairs of chips. Table 1 gives the maximum storage for each chip density and total number of chips installed. A jumper option on the MRV11-C allows this board to respond to both DATI and DATIO bus cycles to allow use with programs using the KEV11 extended instructions on the LSI-11/2 processor.

When used in an LSI-11/2 or LSI-11/23 system, this board may operate in either direct addressing mode or window mapped mode. In addition, the board may optionally provide a system bootstrap window.

## DIRECT MODE OPERATION

When functioning in direct mode, the MRV11-C serves as a high-density replacement for the MRV11-AA or MRV11-BA ROM memory modules. The base address of the direct mode ROM area is assignable on any 8KB boundary from 0 to 248KB (000000<sub>8</sub> to 760000<sub>8</sub>). When operated in this mode, the application program is executed directly from the MRV11-C storage.

## WINDOW MAPPED (PAGED) MODE OPERATION

When window mapped operation is selected, the entire contents of the ROM board are not visible to the LSI-11 address space at any particular point in time. Instead, any two 2KB segments of the ROM can be addressed through two independent windows defined in the LSI-11 system's address space. The association of segments of the ROM board with windows is controlled by a control and status register. Refer to Figure 1 for an example of this operation.

### CSR Definition

Each MRV11-C board uses one 16-bit control and status register located in the system I/O page to determine mapping of ROM

segments into windows in the window mapped mode. The default address for this CSR is 177000<sub>8</sub> (777000<sub>8</sub> in 11/23 systems). The valid address range for CSR's is 177000<sub>8</sub> to 177036<sub>8</sub> (777000<sub>8</sub> to 777036<sub>8</sub> on 11/23 systems). Figure 2 shows the bit assignments for the MRV11-C control and status register.

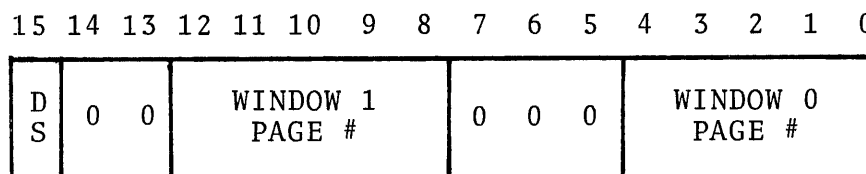


FIGURE 2  
MRV11-C CONTROL AND STATUS REGISTER FORMAT

The CSR contains a 5-bit read/write field for each window. The number stored in this field (0 to 31<sub>10</sub>) selects the desired 2KB region from the MRV11-C board to be associated with the window in question. CSR bits 0 through 4 control the mapping of the low address window, window 0. The low 5 bits of the upper byte, bits 8 through 12, control the mapping of window 1.

The MRV11-C optionally provides a window enable/disable capability. When this option is selected, bit 15 of the CSR is used to enable or disable window response under program control. When bit 15 is a 0, the board will respond to references to the CSR or DATI or DATIO references to either of the windows. When bit 15 is a 1, only the CSR will respond. If the enable/disable option is not selected, bit 15 of the CSR will be read only and always 0. The enable/disable bit has no effect on direct mode addressing or the bootstrap window capability.

The remaining bits in the CSR (bits 5-7 and bits 13-14) are reserved and must always be zero.

### Window Definition

Each MRV11-C board provides a pair of 2KB windows. These windows are always contiguous with each other, and the base address of the window pair may be set to any 4KB boundary in the LSI-11 address



space from 000000<sub>8</sub> to 770000<sub>8</sub>. To maximize the amount of space left for system RAM memory, a default window base of 160000<sub>8</sub> (760000<sub>8</sub> for 11/23 systems) is suggested. Figure 1 shows such a window configuration.

## Using Multiple Boards

Up to 16 MRV11-C boards may be configured in a single system. When multiple boards are present, each board has a unique control and status register address assigned in increasing order from 177000<sub>8</sub> (777000<sub>8</sub> in 11/23 systems). Each board can have a unique 4KB area of the physical address space set aside for its windows, but it is also possible to share one 4KB area of the address space among all MRV11-C boards installed in the system. This is done by using the enable/disable capability discussed earlier. When enable/disable is implemented, the disable bit in the CSR will be set automatically by BINIT on the bus or by execution of the RESET instruction. Therefore, the initial state of the system will have all boards disabled. To access a particular segment of ROM in this multi-board configuration, the programmer first enables the desired board and maps the segment. When access to that segment is completed, the board is again disabled to allow another board to be selected at a future time.

## Sizing Window Mapped Boards

When a board is totally populated (i.e., all 16 PROM sockets are occupied), the mapping values in the CSR will be interpreted modulo the size of the board. Thus, if the board is populated with 2048x8 chips and the programmer selects the 16th 2KB section, the 0th 2KB section will be mapped by the window. If, however, a board is partially populated, an attempt to access the unpopulated area of the board will result in a bus time-out trap.

## BOOTSTRAP WINDOW

An additional optional feature of the MRV11-C board is the capability to respond to the standard PDP-11 bootstrap addresses. When this feature is enabled, an attempt to reference addresses 173000<sub>8</sub> to 173777<sub>8</sub> (773000<sub>8</sub> to 773777<sub>8</sub> in 11/23 systems) will be automatically rerouted to a user-selected 512-byte section of the MRV11-C board. In this way, 512 bytes of the board will be visible in two places in the LSI-11 address space. The highest addressed 512 bytes of any 2KB segment of the board may be selected to correspond to the bootstrap window. This allows custom bootstraps to be added to an LSI-11 system without requiring an additional board to store the bootstrap.

## WINDOW MAPPED MODE APPLICATION EXAMPLES

The window mapped mode may be used in two different ways in LSI-11 application programs. The application can be coded in such a way as to execute directly from the windows, or the window mapped board may be used as a program load device to transfer a stand-alone application program from ROM into RAM memory at system start-up.

### Executing Windowed Programs

Executing directly from MRV11-C windows allows very large program sizes with up to 56KB of RAM on LSI-11/2 systems. However, software to be executed in this mode must be custom designed and must be written in assembly language.

An application designed for windowed execution must have a mechanism for calling a subroutine or transferring control to another routine which is physically located in a presently unmapped section of the windowed ROM board. To accomplish this, we must use a technique different from the standard JSR or JMP instructions. A method for doing this is illustrated in Figure 3. The routine which processes subroutine calls and jumps to other pages must, of course, be located in a section of memory which is not window mapped. To call a subroutine using these capabilities, one would write CALLW0 label rather than JSR PC, label. This would cause the subroutine desired to be mapped into window 0 and the call to be executed. Upon subroutine return which is done with a normal RTS PC instruction, the original mapping would be restored and control would be returned to the calling program unit. Likewise, to invoke a subroutine but have it mapped in window 1, the programmer codes CALLW1 label. Note that the mechanism shown in the figure preserves condition codes from the called routine back to the caller (i.e., routines can return status in the condition codes). Instead of the unconditional jump instruction, the programmer codes JMPW0 label to jump to a routine, mapping it into window 0. Similarly, one can code JMPW1 label to transfer control to a routine which should be mapped into window 1.

To make use of this functionality, the program should be assembled with .ENABL AMA to force absolute addressing in the assembly and at start-up time, a boot routine must be executed (from the MRV11-C boot window or elsewhere) which copies the trap handler routine to RAM memory, if necessary, and initializes the trap vector to contain the address of the trap handling routine and a new status word of all 0's.

The programmer of this type of application must take care not to cross page boundaries without remapping to the next page. If a page boundary is encountered, the JMPW0 or JMPW1 pseudo instructions should be used to get to the beginning of the next page.

## Using Window Mapping As A Program Loader

The MRV11-C in window mapped mode can also be used as a low-cost program load device for stand-alone applications. This allows application programs which cannot be easily segmented into ROM and RAM sections to be loaded from a ROM environment into RAM for execution. To use the MRV11-C in this mode, a bootstrap loader program must be written to copy the contents of the ROM board into the RAM area at power-up. Figure 4 demonstrates such a program designed to load stand-alone images which have been created by the RT-11 LINK utility. (Figure 4 is attached)

It is also possible to load an RSX-11S system image from one or more MRV11-C boards into RAM for execution. The loader required for this process will be the subject of another Micro Note.

# OF CHIPS INSTALLED	CHIP SIZE		
	1024x8	2048x8	4096x8
2	2KB	4KB	8KB
4	4KB	8KB	16KB
6	6KB	12KB	24KB
8	8KB	16KB	32KB
10	10KB	20KB	40KB
12	12KB	24KB	48KB
14	14KB	28KB	56KB
16	16KB	32KB	64KB

TABLE 1

TOTAL STORAGE CAPACITY PER BOARD  
AS A FUNCTION OF SIZE AND NUMBER OF CHIPS

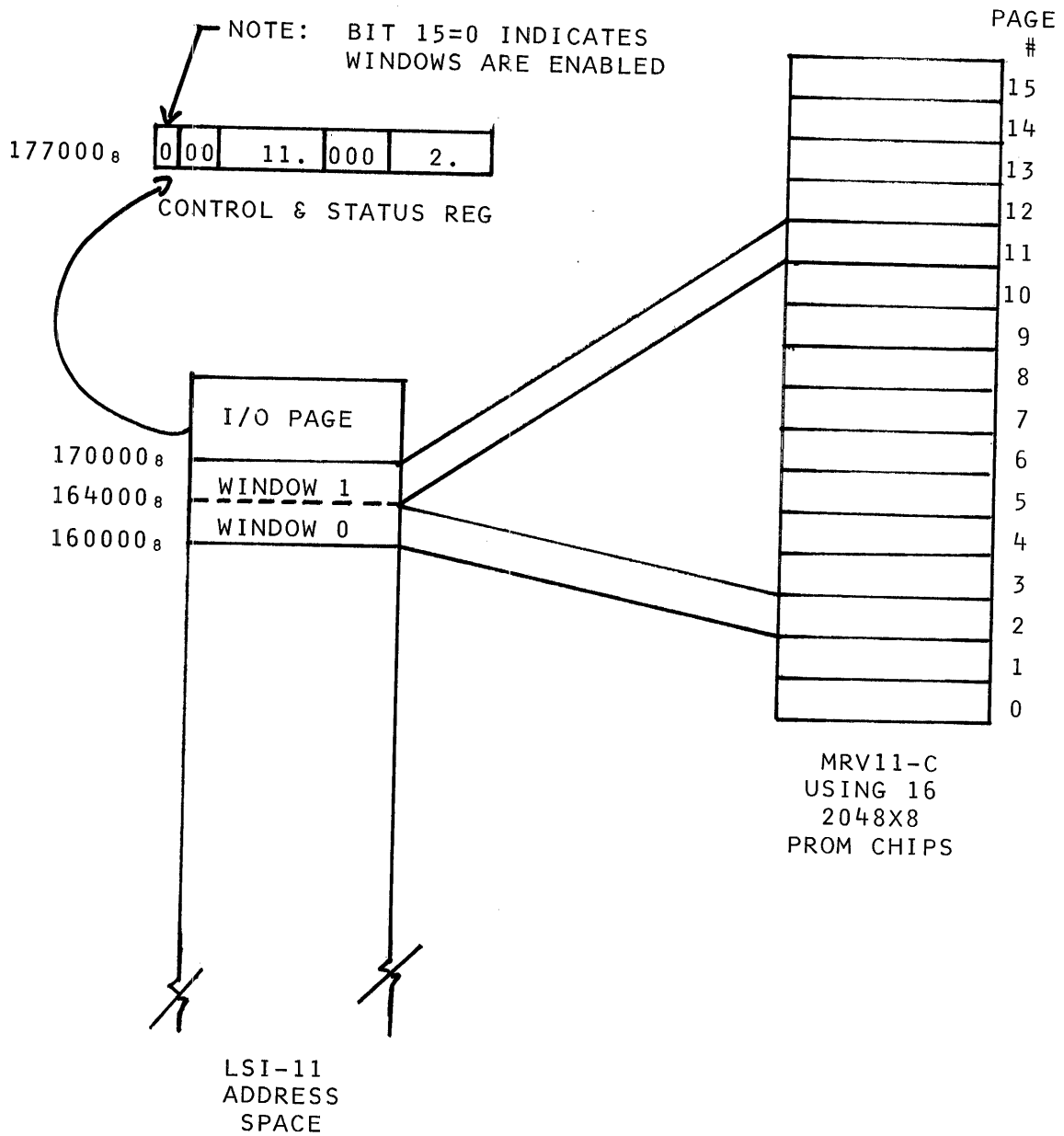


FIGURE 1  
EXAMPLE OF WINDOW-MAPPED OPERATION

WOBASE=160000  
 W1BASE=164000  
 JMPW =1  
 JSRW =0  
 W1 =2  
 W0 =0

NUMBER 061  
 PAGE 8 OF 9

	.MACRO	CALLW0	ADRS
	TRAP	JSRW+W0+<<ADRS/1000>&174>	
	.WORD	WOBASE+<ADRS&3777>	
	.ENDM	CALLW0	
	.MACRO	JMPW0	ADRS
	TRAP	JMPW+W0+<<ADRS/1000>&174>	
	.WORD	WOBASE+<ADRS&3777>	
	.ENDM	JMPW0	
	.MACRO	CALLW1	ADRS
	TRAP	JSRW+W1+<<ADRS/1000>&174>	
	.WORD	W1BASE+<ADRS&3777>	
	.ENDM	CALLW1	
	.MACRO	JMPW1	ADRS
	TRAP	JMPW+W1+<<ADRS/1000>&174>	
	.WORD	W1BASE+<ADRS&3777>	
	.ENDM	JMPW1	
TRPHAN:	MOV	@#MRVCSR, -(SP)	;Save previous mapping
	TST	-(SP)	;Reserve space for adrs
	MOV	R0, -(SP)	;Save caller's register
	MOV	6(SP), R0	;And set R0 to address of TRAP+2
	ADD	#2, 6(SP)	;Update return PC beyond adrs
	MOV	@R0, 2(SP)	;Move adrs (follows TRAP instruction)
	MOV	-(R0), R0	;R0=TRAP instruction itself
	BIC	#177600, R0	;Extract page #, window #, JMP/JSR
	ASR	R0	;Move JMP/JSR to C bit
	ROR	R0	;Place window # in C, JMP/JSR in bit 15
	MOV	#MRVCSR, -(SP)	;Set address of window 0 map bits
	ADC	@SP	;And update based on window #
	MOVB	R0, @(SP)+	;Map new page in selected window
	ROL	R0	;JMP/JSR back to C bit
	MOV	(SP)+, R0	;Restore caller's register
	BCS	1\$	;If JMP, branch to 1\$
	JSR	PC, @(SP)+	;Else JSR to desired routine
	MFPS	4(SP)	;Store returned condition codes in old PS
	MOV	(SP)+, @#MRVCSR	;Restore original mapping
	RTI		;And return after TRAP and adrs
1\$:	MOV	(SP)+, @SP	;If JMP, move adrs
	MOV	(SP)+, @SP	;Up over old (caller's) PC
	RTI		;And go to new location

FIGURE 3

JSR AND JMP WINDOW MAPPED CONTROL ROUTINES

```

MRVCSR=      177000
MRVWIN=      160000

ONEKW=       003777


LOADER:      CLR          @#MRVCSR      ;Enable & Map Low 1K Words
              MOV          @#MRVWIN+50,R5;R5 = RT-11 SAV File High Limit
              CLR          R4           ;Start Copying Into Location 0
1$:           MOV          #MRVWIN,R3   ;Reset to Base of First Window
2$:           MOV          (R3)+, (R4)+ ;Copy One Word Into RAM
              CMP          R4,R5       ;Moved Highest Word in Program?
              BHIS        3$          ;If HIS, Yes
              BIT         #ONEKW,R3    ;Have Reached Next 1KW Boundary?
              BNE        2$          ;If NE, No
              INC         @#MRVCSR     ;Else Map Next 1K in Window 0
              BR         1$          ;And Continue Copying

3$:           MOV          @#40,PC      ;Start at User's Transfer Address

```

FIGURE 4

BOOTSTRAP LOADER FOR STAND-ALONE PROGRAMS IN RT-11 .SAV FORMAT

	NUMBER 062A
	DATE 6 / 14 / 79
	PRODUCT TU58, RL01, RK05, RX02, RX01
	PAGE 1 OF 31
TITLE <u>Bootstraps for TU58, RL01, RK05, RX02, RX01</u>	
DISTRIBUTION <u>Unrestricted</u>	
ORIGINATOR <u>Rich Billig</u>	

THIS MICRO NOTE SUPERSEDES MICRO NOTE #062 ON THE SAME SUBJECT DATED 2/2/79

The attached listings contain the programs stored in the MXV11-A2 bootstrap chips. These programs replace the preliminary versions originally published in Micro Note #062.

Either of these programs can be used in a 256-word bootstrap PROM environment in MRV11-AA, MRV11-C or other ROM modules.

For more details on the operation, please consult the comments in the program listings.



MXV11 TU58 BOOTSTRAP      MACRO V03.02B13-JUN-79  
TABLE OF CONTENTS

2-	1	Definitions and Protocol Equates
4-	1	Memory Test
5-	1	Subroutine to Write, Read, and Verify Memory
5-	25	*****
5-	25	***** HALT AT PC=173076 INDICATES "MEMORY ADDRESS ERROR"
5-	25	*****
6-	1	Memory Data Storage Test
6-	20	*****
6-	20	***** HALT AT PC=173122 INDICATES "BAD MEMORY DATA"
6-	20	*****
6-	35	*****
6-	35	***** HALT AT PC=173140 INDICATES "MEMORY BACKGROUND DATA ERROR"
6-	35	*****
7-	1	MACRO, FORTRAN-Callable Loader Entry
7-	11	*
7-	11	*----> TULOAD ENTRY POINT IS AT ADDRESS 173154
7-	11	*
8-	1	Bootstrap TU58
8-	31	*****
8-	31	***** HALT AT PC=173274 INDICATES "CONTROLLER ERROR - SUCCESS CODE IN RO LOW BYTE"
8-	31	*****
9-	1	Stand-Alone File Loader
9-	23	*****
9-	23	***** HALT AT PC=173374 INDICATES "DESIRED FILE WAS NOT FOUND"
9-	23	*****
9-	40	*****
9-	40	***** HALT AT PC=173440 INDICATES "PROTOCOL ERROR IN READ OPERATION"
9-	40	*****
10-	1	Load Stand-Alone Program File
10-	11	*****
10-	11	***** HALT AT PC=173474 INDICATES "START ADDRESS INVALID"
10-	11	*****
11-	1	Start READ Operation on TU58
13-	1	TU58 Interface I/O Routines

```
1          .TITLE  MXV11 TU58 BOOTSTRAP
2          .ENABL  LC
3
4          ; Edit level 11, made 09-Mar-79 by RRB
5
6          ; November 1978 by RRB
7          ; Copyright (C) 1978
8          ; by Digital Equipment Corporation, Maynard, Massachusetts 01754
9
10         .REPT   0
11
12         000000
13
14         This is a 256-word diagnostic and bootstrap program for LSI-11 systems
15         using the TU58 DECTape II tape cartridge drives.  It performs the following
16         functions:
17
18         1.  On power up, size memory (up to 30K words).
19         2.  Test addressing of memory by writing each location with its
20             address and verifying the contents.
21         3.  Check data retention, writing 1's into a 0's background, and
22             vice-versa.
23         4.  Attempt to bootstrap the TU58 cartridge on unit 0.  If unit 0 fails
24             to function, cycle to unit 1 and try it.  Continue cycling between
25             units 0 and 1 until one correctly reads block 0 of the tape.
26         5.  Check the first word read to see if it is 240(8).  If so, start
27             program execution at location 0 with interrupts disabled (PR7).
28             (This is the standard PDP-11 bootstrap convention.)
29             If the first word is not 240(8), check to see if it is 260(8).
30             If so, this signals a special "stand-alone" program load request
31             to this bootstrap program.  If not 260(8), return to step #4 and
32             cycle to the next unit.
33         6.  If a stand-alone program load request was made (i.e., location 0
34             contains 260(8)), scan the RT-11 directory of the unit to find
35             the file whose name is given in locations 2-6 in block 0 of the
36             cartridge.  The name is represented as three words of RADIX50 data
37             denoting the desired filename and extension.
38         7.  If the file is found in the directory, load it as an RT-11 .SAV
39             image file, and start its execution.
40
41         The bootstrap ROM also provides a MACRO or FORTRAN-callable entry point
42         which can be used to "chain" from one stand-alone program to another.
43         The format of the FORTRAN CALL is:
44
45         CALL TULOAD( ifile )
46
47         where "ifile" is a four-element INTEGER array containing:
48
49         ifile(1) = the unit # of the TU58 drive to use (binary 0 or 1)
50         ifile(2) = the RADIX50 code for the first 3 characters of the filename
51         ifile(3) = the RADIX50 code for the last 3 characters of the filename
52         ifile(4) = the RADIX50 code for the file extension
53
54         Calling the entry point TULOAD with the appropriate FORTRAN-format argument
55         list will cause the bootstrap to search the directory of the indicated tape
56         unit for the file specified, and if found, load it into memory and start its
57         execution.
58
59         .ENDR
```

```

1          .SBTTL  Definitions and Protocol Equates
2
3          ; Absolute address definitions
4
5          000002      FILNAM = 000002          ;Address of RAI50 filename for stand-alone
6                                     ; program loadins
7          001000      DIRBUF = 001000        ;Start of 512. word buffer used for RT-11
8                                     ; directory operations in stand-alone loadins
9
10         ; TU58 Address definitions
11
12         176500      TI$CSR = 176500         ;DL receiver control and status
13         176502      TI$BFR = 176502         ;DL receiver data buffer
14         176504      TO$CSR = 176504         ;DL transmitter control and status
15         176506      TO$BFR = 176506         ;DL transmitter data buffer
16
17         ; TU58 Radial Serial Protocol codes
18         ; Flag Byte Definitions:
19
20         000001      R$$DAT = 'B<00001>     ;Data message flag
21         000002      R$$CTL = 'B<00010>     ;Control message flag
22         000004      R$$INT = 'B<00100>     ;Initialize flag
23         000020      R$$CON = 'B<10000>     ;Continue flag
24         000023      R$$XOF = 'B<10011>     ;XOFF
25
26         ; Control packet operation codes:
27
28         000000      R$NOP  = 0.             ;No-operation
29         000001      R$INIT = 1.             ;Initialize
30         000002      R$READ = 2.             ;Read operation
31         000003      R$WRIT = 3.             ;Write operation
32         000004      R$COMP = 4.             ;Compare (NOP on TU58)
33         000005      R$POSI = 5.             ;Position operation
34         000006      R$ABRT = 6.             ;Abort (NOP on TU58)
35         000007      R$DIAG = 7.             ;Diagnose
36         000010      R$GETS = 8.             ;Get status
37         000011      R$SETS = 9.             ;Set status (NOP on TU58)
38         000012      R$GETC = 10.            ;Get characteristics
39         000013      R$SETC = 11.            ;Set characteristics (NOP on TU58)
40         000100      R$END  = 'B<01000000>   ;*END message
41
42         ; END packet success codes:
43
44         000000      S$NORM = 0.             ;Normal success
45         000001      S$RETR = 1.             ;Success but with retries
46         177776      S$PART = -2.            ;Partial operation (end of medium)
47         177770      S$UNIT = -8.            ;Invalid unit number
48         177767      S$CART = -9.            ;No cartridge
49         177765      S$WPRT = -11.           ;Cartridge write protected
50         177757      S$DCHK = -17.           ;Data check error
51         177740      S$SEEK = -32.           ;Seek error (block not found)
52         177737      S$MOTR = -33.           ;Motor stopped
53         177720      S$OPCD = -48.           ;Invalid operation code
54         177711      S$RECN = -55.           ;Invalid record number

```

```

1          ; RT-11 Directory Structure Definitions
2
3          001000          SEGALD = DIRBUF          ;Number of segments allocated
4          001002          NXTSEG = DIRBUF+2        ;Number of next logical segment
5          001004          HGHSEG = DIRBUF+4        ;Highest segment in use
6          001006          XTRBYT = DIRBUF+6        ;Number of extra bytes per entry
7          001010          STRBLK = DIRBUF+10       ;Starting block# for files in this segment
8
9          000016          ENTSIZ = 7*2            ;Size of a directory entry
10         000010          D.FLEN = 10              ;Offset to file length in entry
11         000400          TENTAS = 000400         ;Flas for tentative file entry
12         001000          EMPTY$ = 001000         ;Flas for empty area entry
13         002000          PERMF$ = 002000         ;Flas for permanent file
14         004000          ENDSG$ = 004000         ;Flas for end of segment
15
16         ; RT-11 System Communications Area Definitions
17
18         000040          RT$STA = 000040         ;Start address for program
19         000042          RT$ISP = 000042         ;Initial stack pointer
20         000044          RT$JSW = 000044         ;Job status word
21         000046          RT$USR = 000046         ;USR load address
22         000050          RT$HGH = 000050         ;Job high memory limit
23         000052          RT$EMT = 000052         ;(Byte) EMT error code
24         000053          RT$UER = 000053         ;(Byte) User error code
25         000054          RT$RMN = 000054         ;Base address of resident monitor
26         000056          RT$FCH = 000056         ;(Byte) Console fill character
27         000057          RT$FCT = 000057         ;(Byte) Console fill count
28
29         ; Error macro definition
30
31         .MACRO ERROR TEXT
32             HALT
33         .IRP PC$,\,
34         .SBTTL *****
35         .SBTTL ***** HALT at PC='PC$ indicates ''TEXT''
36         .SBTTL *****
37         .ENDR
38         .ENDM ERROR
  
```

```

1          .SBTTL Memory Test
2
3 000000          .ASECT
4          .IIF NDF ORIGIN, ORIGIN=173000 ;Set default orisin to besinnins of boot
5          .=ORIGIN                      ;Orisin to besinnins of bootstrap ROM
6
7          ; Start of Memory Diagnostics
8          ;
9
10 173000 010701 MEM:  MOV      PC,R1          ;Go size memory
11 173002 000402      BR      MEMSIZ        ;Subroutine call
12 173004 005003      CLR      R3          ;Set test data = 0
13 173006 000423      BR      MEMDO        ;Go do memory test
14
15
16          ; Memory sizing subroutine (called with R1 convention)
17          ; Sets R4 to highest writable memory address, R2=2
18
19 173010 012700 000340 MEMSIZ: MOV      #340,R0          ;Assure no interrupt enable on restart
20 173014 106400      MTPS     R0          ; by setting priority 7
21 173016 000005      RESET    ;Reset external bus
22 173020 012702 000006      MOV      #6,R2          ;R2 -> PS word in timeout vector
23 173024 106712      MFPS     @R2          ;Set new PS priority to 7
24 173026 012742 173046      MOV      #2*-ORIGIN+173000,-(R2) ; and new PC to escape label
25 173032 005004      CLR      R4          ;Set initial start address
26 173034 012706 001000      MOV      #1000,SP        ;And initial stack pointer
27 173040 011414      1$:      MOV      @R4,@R4        ;Try to read and write each memory location
28 173042 005724      TST      (R4)+        ;If no trap on DATI or DATO, skip to next
29 173044 000775      BR      1$          ;And loop until timeout occurs
30
31 173046 034442      2$:      BIT      -(R4),-(R2)        ;Set R4 -> highest writable adrs, R2=2
32 173050 010406      MOV      R4,SP        ;And set stack at top of memory
33 173052 000161 000002      JMP      2(R1)          ;Return to caller

```

```

1          .SBTTL  Subroutine to Write, Read, and Verify Memory
2
3          ; Subroutine to Write, Read, and Verify Memory
4          ; Enter with:
5          ;       R3 = 0
6          ;       R4 = High address for verification
7          ;
8          ; Calling Sequence:
9          ;       MOV    PC,R1
10         ;       BR    MEMDO
11         ;
12         ; Subroutine only returns if no errors are detected.
13         ; The contents of R2 are destroyed.
14         ; The test consists of a memory address and data storage test.
15         ; First we write all of memory with its address and then read
16         ; and verify memory.
17
18 173056 005002  MEMDO: CLR    R2          ;Copy starting address
19 173060 010212  ADDRT: MOV    R2,@R2      ;Store address at address
20 173062 005722          TST    (R2)+    ;And skip to next
21 173064 020204          CMP    R2,R4      ;Finished with all addresses?
22 173066 101774          BLOS  ADDRT      ;If LOS no
23 173070 024202  CHECK:  CMP    -(R2),R2    ;Yes, check data
24 173072 001401          BEQ    ADCONT     ;If EQ, no comparison error
25 173074          MEMERA: ERROR  <Memory address error>
26         ; Expected data is in R2; bad data is pointed to by adrs in R2.
27         ; Type 'P' to continue test.
28
29 173076 005702  ADCONT: TST    R2          ;Have we finished the check?
30 173100 001373          BNE    CHECK      ;If NE, no

```

```

1          .SBTTL  Memory Data Storage Test
2
3          ; Memory data storage test.
4          ;
5          ; The steps of this test are:
6          ; 1. Fill memory with zeroes.
7          ; 2. Walk an all 1's word through memory & verify each location.
8          ; 3. Fill memory with ones.
9          ; 4. Walk an all 0's word through memory & verify each location.
10         ; By performing these steps all bit positions are checked for 0/1
11         ; storage and the sense amps are stressed in the semiconductor memories.
12
13 173102 010322  MEMT:  MOV    R3,(R2)+      ;Move background data to memory
14 173104 020204          CMP    R2,R4      ;Done with desired area?
15 173106 101775          BLOS  MEMT      ;If LOS, no
16 173110 005103  WALK:  COM    R3          ;Complement test data
17 173112 074342          XOR    R3,-(R2)     ;Use background & test data to set all 1's
18 173114 005112          COM    @R2      ;Check for all 1's (tests previous bcks)
19 173116 001401          BEQ    DCONT     ;If EQ, data is good
20 173120          MEMERD: ERROR  <Bad memory data>
21         ; Expected data is in R3; bad data is pointed to by adrs in R2.
22         ; Type "P" to continue test.
23
24 173122 005103  DCONT:  COM    R3          ;Get previous background data
25 173124 074312          XOR    R3,@R2     ;Restore background data for current location
26 173126 005702          TST   R2          ;Done with one pass?
27 173130 001367          BNE   WALK      ;If NE, no
28
29         ; At this point, the pattern has been walked through all of memory.
30         ; Go back through examining background data to make sure it was not
31         ; effected by modification of other locations.
32
33 173132 020322  BACK:  CMP    R3,(R2)+      ;Is background still valid?
34 173134 001401          BEQ    BGCNT     ;If EQ, yes
35 173136          MEMERB: ERROR  <Memory background data error>
36
37 173140 020204  BGCNT:  CMP    R2,R4      ;Scanned all of background yet?
38 173142 101773          BLOS  BACK      ;If LOS, no - continue with next word
39 173144 005002          CLR   R2        ;Else restore R2 to low memory limit
40 173146 005103          COM   R3        ;Flip to other pattern
41 173150 001354          BNE   MEMT      ;If NE, test not complete yet
42 173152 000411          BR   BOOT      ;Go kick the boot

```

```

1          .SBTTL  MACRO, FORTRAN-Callable Loader Entry
2
3          ; The followins entry point, callable from MACRO or FORTRAN programs,
4          ; will autoload and start an executable file whose name is specified
5          ; by the FORTRAN-style arsument list pointed to by R5.
6
7          .IRP PC$,\
8          .SBTTL  *
9          .SBTTL  *----> TULOAD entry point is at address 'PC$
10         .SBTTL  *
11         .ENDR
12
13 TULOAD::
14 173154      MOV      PC,R1          ;Subroutine call
15 173154 010701  BR      MEMSIZ      ; to size memory
16 173156 000714      MOV      2(R5),R5    ;R5 -> four element array
17 173160 016505 000002  MOV      (R5)+,@R4    ;Set unit # on top of new stack
18 173164 012514      MOV      (R5)+,(R2)+    ;Copy filename.ext
19 173166 012522      MOV      (R5)+,(R2)+    ; in RAD50 to locations
20 173170 012522      MOV      @R5,@R2      ; 2,4, and 6 of memory
21 173172 011512      BR      STANDB      ;And so load file
21 173174 000446

```



```

1          .SBTTL  Bootstrap TU58
2
3          .ENABL  LSB
4 173176  012701  176504      BOOT:  MOV    #T0%CSR,R1      ;R1 -> output CSR for TU58 serial line
5 173202  005303              DEC    R3                ;Set R3 = 177777 (Two RUBOUT characters)
6 173204  005211              INC    @R1              ;Start transmitting BREAK to TU58
7 173206  004767      000514  CALL   CHROUT          ;Send eight RUBOUTs
8 173212  105711      1$:    TSTB   @R1                ;Is transmitter ready again yet?
9 173214  100376              BPL   1$                ;If PL no - wait
10 173216  005011              CLR   @R1              ;Else stop sending BREAK now
11 173220  012703              MOV   (PC)+,R3        ;Get two INIT commands for TU58
12 173222         004         004  .BYTE  R$$INT,R$$INT
13 173224  004715              CALL  @R5              ;And transmit them
14 173226  005741              TST  -(R1)            ;Dump any garbage char in TI$BUF
15 173230  105737      176500  2$:    TSTB   @#TI$CSR      ;Is a character available from the TU58?
16 173234  100375              BPL  2$                ;If PL, no - wait in loop
17 173236  121127      000020  CMPB  @R1,#R$$CON     ;If so, was it a CONTINUE flag?
18 173242  001075              BNE  LODERR           ;If NE, no - ERROR
19
20          ; TU58 is now initialized. Prepare to read block #0.
21          ;
22 173244  005416              NEG   @SP              ;Set unit=1 in low byte, unit=0 in high
23 173246  000316              SWAB @SP              ;Switch units
24 173250              REBOOT:
25 173250  005000      4$:    CLR   R0                ;Reference label for retries
26 173252  012701      001000  MOV   #512,,R1        ;Block number = 0
27 173256  004767      000230  CALL  READZU          ;Byte count = one block
28 173262  100005              BPL  5$                ;Attempt to read the block
29 173264  120027      177767  CMPB  R0,#S$CART     ;If PL, read was successful
30 173270  001766              BEQ  3$                ;Did it fail because no cartridge present?
31 173272              ERROR  <Controller error - success code in R0 low byte>
32 173274  000641              BR   MEM              ;If EQ yes - so try other drive
33
34 173276              5$;;  CLR   R5                ;Proceed will restart memory test
35 173276  021527      000240  CMP   @R5,#240        ;Point to address 0 (after READ, R5=0)
36 173302  001475              BEQ  START            ;Did we read a valid bootstrap block?
37 173304  022527      000260  CMP   (R5)+,#260     ;Go start execution (Note: R5 = 0 here!!!)
38 173310  001356              BNE  3$                ;Is this a stand-alone load tape?
39          .DSABL  LSB
40 173312      STANDB:

```

```

1          .SETTL  Stand-Alone File Loader
2
3          ; This routine loads stand-alone programs (assumed to be in RT-11 .SAV
4          ; file format) from an RT-11 file structured TU58 cartridge.  It is
5          ; invoked if the first word in block 0 of the cartridge is a 260.
6
7 173312 012700 000001  STANDB: MOV      #1,R0          ;Set directory segment #1
8 173316 006300          1$:  ASL      R0              ;Two blocks per segment
9 173320 022020          CMP      (R0)+,(R0)+        ;Add 4 to R0, as directory starts in block#6
10 173322 012701 002000  MOV      #1024.,R1         ;Prepare to read two blocks
11 173326 012704 001000  MOV      #DIRBUF,R4        ;Into the directory buffer
12 173332 004767 000156  CALL     READU           ;Read the segment
13 173336 100437          BMI     LODERR          ;If MI, read was unsuccessful
14 173340 012704 001010  MOV      #STRBLK,R4       ;Else prepare to pick up starting block
15 173344 012400          MOV      (R4)+,R0         ;R0 = starting block for files
16 173346 010403          2$:  MOV      R4,R3          ;Save pointer to current entry
17 173350 022724 002000  CMP      #PERMF$, (R4)+   ;Is this a permanent file?
18 173354 001410          BEQ     4$              ;If EQ, yes - so check if it matches
19 173356 022744 004000  CMP      #ENDSG$, -(R4)   ;Else is this the end-of-segment marker?
20 173362 001015          BNE     5$              ;If NE, no - so skip this entry
21 173364 013700 001002  MOV      @#NXTSEG,R0      ;Else set number of next segment
22 173370 001352          BNE     1$              ;If NE, there is one - so read it
23 173372          3$:  ERROR   <Desired file was not found>
24 173374 000776          BR      3$              ;Cannot continue!
25
26 173376 012705 000002  4$:  MOV      #FILNAM,R5    ;Point to RAD50 name of desired file
27 173402 022425          CMP      (R4)+,(R5)+     ;Check file name, first word
28 173404 001004          BNE     5$              ;If NE not desired file
29 173406 022425          CMP      (R4)+,(R5)+     ;...Check second word of filename
30 173410 001002          BNE     5$              ;If NE not desired one
31 173412 022425          CMP      (R4)+,(R5)+     ;...Finally, check extension
32 173414 001413          BEQ     LOAD            ;If EQ, got it - so load this one into memory
33 173416 010304          5$:  MOV      R3,R4          ;Get entry pointer back
34 173420 062704 000010  ADD     #D.FLEN,R4        ;Advance to file size of entry
35 173424 062400          ADD     (R4)+,R0         ;Update current file base
36 173426 022424          CMP      (R4)+,(R4)+     ;And skip to next file entry
37 173430 063704 001006  ADD     @#XTRBYT,R4       ;Plus any extra bytes in each entry
38 173434 000744          BR      2$              ;Continue file search
39
40 173436          LODERR: ERROR   <Protocol error in READ operation>
41 173440 000167 177334  JMP     MEM              ;If PROCEED, try again

```

```

1
2                                .SBTTL Load Stand-Alone Program File
3 173444 011401                LOAD:  MOV    @R4,R1          ;R1 = size of file in blocks
4 173446 000301                SWAB   R1                  ; * 256. = word count
5 173450 006301                ASL    R1                  ; * 2 = byte count
6 173452 004767 000034        CALL   READZU         ;Read the program file into memory
7 173456 100767                BMI   LODERR         ;If MI, error in read
8 173460 013705 000040        MOV    @#RT$STA,R5   ;Get program start adrs
9 173464 032705 000001        BIT    #1,R5         ;Is adrs even?
10 173470 001402                BEQ   START         ;If EQ yes - okay
11 173472                                1$:  ERROR  <Start address invalid>
12 173474 000776                BR    1$            ;Cannot continue from here
13
14 173476 112600                START: MOVB  (SP)+,R0   ;Get unit number booted
15 173500 012701 176500        MOV    #TI$CSR,R1   ;Pass the CSR address
16 173504 013706 000042        MOV    @#RT$ISP,SP  ;Load program's stack pointer
17 173510 000115                JMP    @R5          ;Go start program execution

```

```

1          .SBTTL Start READ Operation on TU58
2
3          ; Starts a read operation on the TU58 by transmitting a command packet
4          ;
5          ; Inputs:
6          ;     R0 = starting block # for transfer
7          ;     R1 = byte count for transfer
8          ;     R2 = unit number
9          ;     R4 = address of buffer to receive data
10         ; Outputs:
11         ;     R0, R1, R2 unchanged
12         ; Destroys:
13         ;     R3, R4, R5
14
15 173512 005004      READZU: CLR      R4          ;Set buffer address = 000000
16 173514 116602 000002 READU: MOVB   2(SP),R2      ;And set unit number from stack
17 173520 010446      READ:  MOV    R4,-(SP)    ;Save buffer address
18 173522 005004      CLR      R4          ;Init checksum
19 173524 012703 005002 MOV    #10,*400+R4$CTL,R3 ;Set command flag and length
20 173530 004767 000176 CALL   CH2OUT      ;Output two chars and set R5
21 173534 012703 000002 MOV    #R$READ,R3   ;Send read command and modifier=0
22 173540 004715      CALL   @R5
23 173542 010203      MOV    R2,R3          ;Then unit number and switches=0
24 173544 004715      CALL   @R5
25 173546 005003      CLR      R3          ;Plus a zero sequence number
26 173550 004715      CALL   @R5
27 173552 010103      MOV    R1,R3          ;Followed by the byte count
28 173554 004715      CALL   @R5
29 173556 010003      MOV    R0,R3          ;And the block number
30 173560 004715      CALL   @R5
31 173562 010403      MOV    R4,R3          ;Finally, transmit the checksum
32 173564 004715      CALL   @R5

```

```

1          ;
2          ; Now ready to accept data messages from the TU58
3          ;
4 173566 012600          MOV      (SP)+,R0          ;R0 -> data buffer
5          ;          CLC          ;(CH2OUT leaves C clear)
6 173570 006001          ROR      R1          ;R1 = word count for transfer
7 173572 004767 000106  1$: CALL      5$          ;Get first word of packet
8 173576 122703 000001  CALL      5$          ;Is this indeed a data message?
9 173602 001017          BNE      3$          ;If NE no - may be END message
10 173604 105003         CLRB     R3          ;Else clear flass
11 173606 000303         SWAB     R3          ;Move packet byte count to low byte
12 173610 106003         RORB     R3          ;And convert to word count
13 173612 160301         SUB      R3,R1          ;Remove from transfer count
14 173614 010305         MOV      R3,R5          ;And copy for loop counter
15 173616 004767 000072  2$: CALL      7$          ;Get next two words
16 173622 010320         MOV      R3,(R0)+          ;Store in buffer
17 173624 077504         SOB      R5,2$          ;Loop for entire data message
18 173626 004767 000040  CALL      4$          ;Get checksum and compare
19 173632 005701         TST      R1          ;Have all data records been transferred?
20 173634 001356         BNE      1$          ;If NE no
21 173636 004767 000042  CALL      5$          ;And set prospective END packet start
22 173642 004767 000046  3$: CALL      7$          ;Get opcode/success bytes of END packet
23 173646 122703 000100  CMPB     #R$END,R3          ;Is this an END packet?
24 173652 001271         BNE      LODERR          ;If NE no - abort transfer
25 173654 010300         MOV      R3,R0          ;Save success code in R0
26 173656 004767 000026  CALL      6$          ;Read remainder of END packet
27 173662 004767 000004  CALL      4$          ;And check its checksum
28 173666 000300         SWAB     R0          ;Set CC's on success code of transfer
29 173670 000207         RETURN          ;Return to caller
30
31 173672 004767 000060  4$: CALL      CH2IN          ;Get two checksum bytes
32 173676 020403         CMP      R4,R3          ;Does it match calculated value?
33 173700 001256         BNE      LODERR          ;If NE no - ERROR
34 173702 000207         RETURN          ;Else return with success
35
36 173704 005004         5$: CLR      R4          ;Init checksum
37 173706 000402         BR      7$          ;And set the first word
38
39 173710 004717         6$: CALL     @PC          ;Read 4 words
40 173712 004717         CALL     @PC
41 173714 004767 000036  7$: CALL     CH2IN          ;Read next two bytes
42 173720 060304         ADD     R3,R4          ;Add into checksum
43 173722 005504         ADC     R4          ; with end-around carry
44 173724 000207         RETURN          ;And back to caller

```

```

1          .SBTTL  TU58 Interface I/O Routines
2
3          ; CH2OUT -- Write two bytes to the TU58
4          ;
5          ; Writes two bytes to interface and updates checksum.
6          ;
7          ; Inputs:
8          ;   R3 = two bytes to be output; low byte first
9          ;   R4 = current checksum word
10         ; Outputs:
11         ;   R3 unchanged
12         ;   R4 updated to new checksum
13         ;   R5 pointing to CH2OUT routine for easier future CALLs
14
15 173726 004717  CH2OUT: CALL  @PC          ;Entry point to output 8 characters
16 173730 004717          CALL  @PC
17 173732 010705  CH2OUT: MOV  PC,R5          ;Set R5 to following routine adrs
18 173734 060304          ADD   R3,R4          ;Update checksum word
19 173736 005504          ADC   R4              ; with end-around carry
20 173740 004717          CALL  @PC          ;Repeat for both characters
21 173742 105737 176504 1$:  TSTB  @#TO$CSR        ;Is interface ready for output?
22 173746 100375          BPL   1$              ;If PL no - wait
23 173750 110337 176506          MOVB R3,@#TO$BFR      ;Else transmit character to TU58
24 173754 000407          BR   CHRET         ;Merge with other routine to return
25
26         ; CH2IN -- Read two bytes from the TU58
27         ; CHIN  -- Read a single byte from the TU58
28         ;
29         ; Inputs:
30         ;   none.
31         ; Outputs:
32         ;   R3 = character(s) read
33
34 173756 004717  CH2IN: CALL  @PC          ;Read two, not one
35 173760 105003  CHIN:  CLRB  R3          ;And zero out space for new one
36 173762 105737 176500 1$:  TSTB  @#TI$CSR        ;Is a character available?
37 173766 100375          BPL   1$              ;If PL no
38 173770 153703 176502          BISE  @#TI$BFR,R3      ;Else set into register
39 173774 000303  CHRET: SWAB  R3          ;Move current character over
40 173776 000207          RETURN        ;And return to caller

```

MXV11 TU58 BOOTSTRAP    MACRO V03.02B13-JUN-79 PAGE 14  
TU58 INTERFACE I/O ROUTINES

26

000001

.END

ADCONT 173076	HGHSEG= 001004	RT\$EMT= 000052	R\$NDF = 000000	S\$MOTR= 177737
ADDRT 173060	LOAD 173444	RT\$FCH= 000056	R\$PDSI= 000005	S\$NORM= 000000
BACK 173132	LODERR 173436	RT\$FCT= 000057	R\$READ= 000002	S\$OPCD= 177720
BGCONT 173140	MEM 173000	RT\$HGH= 000050	R\$SETC= 000013	S\$PART= 177776
BOOT 173176	MEMDO 173056	RT\$ISP= 000042	R\$SETS= 000011	S\$RECN= 177711
CHECK 173070	MEMERA 173074	RT\$JSW= 000044	R\$WRIT= 000003	S\$RETR= 000001
CHIN 173760	MEMERB 173136	RT\$RMN= 000054	R\$\$CDN= 000020	S\$SEEK= 177740
CHRET 173774	MEMERD 173120	RT\$STA= 000040	R\$\$CTL= 000002	S\$UNIT= 177770
CH2IN 173756	MEMSIZ 173010	RT\$UER= 000053	R\$\$DAT= 000001	S\$WPRT= 177765
CH2OUT 173732	MEMT 173102	RT\$USR= 000046	R\$\$INT= 000004	TENTA\$= 000400
CH8OUT 173726	NXTSEG= 001002	R\$ABRT= 000006	R\$\$XOF= 000023	TI\$BFR= 176502
DCONT 173122	ORIGIN= 173000	R\$CMP= 000004	SEGALO= 001000	TI\$CSR= 176500
DIRBUF= 001000	PERMF\$= 002000	R\$DIAG= 000007	STANDB 173312	TO\$BFR= 176506
D.FLEN= 000010	READ 173520	R\$END = 000100	START 173476	TO\$CSR= 176504
EMPTY\$= 001000	READU 173514	R\$GETC= 000012	STRBLK= 001010	TULOAD 173154 G
ENDSG\$= 004000	READZU 173512	R\$GETS= 000010	S\$CART= 177767	WALK 173110
ENTSIZ= 000016	REBOOT 173250	R\$INIT= 000001	S\$DCHK= 177757	XTRBYT= 001006
FILNAM= 000002				

. ABS. 174000 000  
 000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 1639 WORDS ( 7 PAGES)  
 DYNAMIC MEMORY AVAILABLE FOR 47 PAGES  
 ,LP:TU58BT=DK:TU58BT/C



2-	1	Controller Definitions
4-	1	Memory Test
4-	42	-----> HALT AT PC=173056 INDICATES "MEMORY ADDRESS ERROR"
5-	1	Memory Data Storage Test
5-	20	-----> HALT AT PC=173102 INDICATES "BAD MEMORY DATA"
6-	1	RL01 Bootstrap
6-	47	-----> HALT AT PC=173270 INDICATES "RL BOOT FAILURE"
7-	1	RK05 Bootstrap
8-	1	MRV11-C Bootstrap
9-	1	RX01/RX02 Bootstrap
10-	1	Miscellaneous Subroutines

```
1          .TITLE  UNIVERSAL DISK BOOTSTRAP
2          .ENABL  LC
3
4          ; Edit level 08 made 01-Mar-79 by RRB
5
6          ; October 1978 by RRB
7          ; Copyright (C) 1978
8          ; by Digital Equipment Corporation, Maynard, Massachusetts 01754
9
10
11          000000          .REPT  0
12
13          This is a 256 word bootstrap program designed to handle all disks which are
14          available for the LSI-11 bus.  It will automatically search for controllers
15          for the various disks (in a predefined order) and bootstrap the first such
16          device which is found and is operable.
17
18          The bootstrap sequence is as follows:
19
20          1.  Size read/write memory, to a maximum of 30K words.
21
22          2.  Perform a memory addressing test, writing each location with its address
23              and verifying the contents.
24
25          3.  Exercise memory data storage capabilities, moving a 1's pattern through a
26              background of 0's and a 0's pattern through a background of 1's, to test
27              all read/write memory locations for independence and retention.
28
29          4.  Check for presence of RLV11 controller.  If not found, proceed to Step #6.
30
31          5.  Attempt to bootstrap RLO1 unit #0.  If there is no such drive,
32              or if no cartridge is present, the cover is open, or the drive is spinning
33              down, proceed to Step #6.
34
35          6.  Check for presence of RKV11 controller.  If not found, proceed to Step #8.
36
37          7.  Attempt to bootstrap each RK05 unit in sequence (0,1,...,7) until a unit
38              is found which is ready and readable.  If no such unit exists, proceed to
39              Step #8.  (This step is preceded by a min. 8 second wait loop to allow
40              sufficient spin-up time.)
41
42          8.  Check for presence of RXV11 or RXV21 controller in the system.  If none
43              exists, proceed to Step #10.
44
45          9.  Wait for min. 2 seconds to allow drive spin-up, then attempt to
46              bootstrap unit 0 of the floppy disk, at the density of the media
47              present in the drive at the time.  If the drive is not ready or does not
48              contain a bootable medium, go to the other unit and try it.  Continue
49              looping between units 0 and 1 until one is found to bootstrap.
50
51          10. Check for the presence of an MRV11-C board with paging enabled.  If not
52              found, proceed to step #4.
53
54          11. Load the first 256 words from the MRV11-C into memory and execute them,
55              if the first word is NOP (000240).  Else proceed to step #4.
56
57          .ENBR
```

```

1          .SBTTL  Controller Definitions
2
3          ; RL01 (RLV11) Register Definitions
4
5          174400      RLCS=  174400      ;Control and Status
6          174402      RLBA=  RLCS+2      ;Bus Address
7          174404      RLDA=  RLBA+2      ;Disk Address
8          174406      RLMP=  RLDA+2      ;Multipurpose Register
9
10         ; RL01 Function Definitions
11
12         000000      RL$NOP= 0*2        ;No-Operation (Maintenance on RLV11)
13         000002      RL$WCK= 1*2        ;Write Check
14         000004      RL$GST= 2*2        ;Get Status (and Reset)
15         000006      RL$SEK= 3*2        ;Seek Cylinder and Select Head
16         000010      RL$RHD= 4*2        ;Read Header
17         000012      RL$WD=  5*2        ;Write Data
18         000014      RL$RD=  6*2        ;Read Data
19         000016      RL$RDN= 7*2        ;Read Data With No Header Check
20
21         ; RL01 Miscellaneous Definitions
22
23         000001      RL$$DR= 000001     ;'Drive Ready' bit in RLCS
24         000013      RLG$RS= 000013     ;Reset and Get Status for RLDA in RL$GST
25         177730      RL$$ER= ^C<000047> ;Mask for Cover Open and State in error bits
26         000006      RLE$UH= 6          ;'Unload Heads' in State info
27         000177      RL$$CA= ^C<177600> ;Mask for cylinder address in header data
28
29         ; RK05 (RKV11) Controller Registers
30
31         177400      RKDS=  177400      ;Drive Status
32         177402      RKER=  177402      ;Error
33         177404      RKCS=  177404      ;Control and Status
34         177406      RKWC=  177406      ;Word Count
35         177410      RKBA=  177410      ;Bus Address
36         177412      RKDA=  177412      ;Disk Address
37         177416      RKDB=  177416      ;Data Buffer
38
39         ; RK05 (RKV11) Function Codes (Plus GO bit)
40
41         000001      RK$CRS= 0*2+1      ;Control Reset
42         000003      RK$WRT= 1*2+1      ;Write
43         000005      RK$RED= 2*2+1      ;Read
44         000007      RK$WCK= 3*2+1      ;Write Check
45         000011      RK$SEK= 4*2+1      ;Seek
46         000013      RK$RCK= 5*2+1      ;Read Check
47         000015      RK$DRS= 6*2+1      ;Drive Reset
48         000017      RK$WLK= 7*2+1      ;Write Lock

```

```

1          ; RX01/RX02 (RXV11,RXV21) Resister Definitions
2
3          177170      RXCS=      177170          ;Control and Status
4          177172      RXDB=      RXCS+2          ;Data Buffer
5
6          ; RX Control and Status Bits
7
8          100000      RX$$ER= 100000          ;Error
9          040000      RX$$IN= 040000          ;Initialize controller
10         030000      RX$$XA= 030000          ;Extended address bits
11         004000      RX$$02= 004000          ;1 if RX02 or RX03; 0 if RX01
12         003000      RX$$XX= 003000          ;Unused bits
13         000400      RX$$DE= 000400          ;Density (1=double,0=sinsle)
14         000200      RX$$TR= 000200          ;Transfer function
15         000100      RX$$IE= 000100          ;Interrupt enable
16         000040      RX$$DN= 000040          ;Done
17         000020      RX$$UN= 000020          ;Unit select
18         000016      RX$$FN= 000016          ;Function select
19         000001      RX$$G0= 000001          ;G0
20
21         ; RX Function Codes (in RX$$FN) with G0 bit preset
22
23         000001      RX$FIL= 0*2+RX$$G0      ;Fill buffer
24         000003      RX$EMP= 1*2+RX$$G0      ;Empty buffer
25         000005      RX$WRT= 2*2+RX$$G0      ;Write sector
26         000007      RX$RED= 3*2+RX$$G0      ;Read sector
27         000011      RX$STD= 4*2+RX$$G0      ;Set media density
28         000013      RX$RST= 5*2+RX$$G0      ;Read status
29         000015      RX$WDD= 6*2+RX$$G0      ;Write sector with deleted data
30         000017      RX$REC= 7*2+RX$$G0      ;Read error code
31
32         ; RX Error Codes
33
34         000400      RXE$UN= 000400          ;Unit selected
35         000200      RXE$DR= 000200          ;Drive ready
36         000100      RXE$DD= 000100          ;Deleted data
37         000040      RXE$DN= 000040          ;Drive density
38         000020      RXE$DE= 000020          ;Density error
39         000004      RXE$ID= 000004          ;Initialize done
40         000001      RXE$CR= 000001          ;CRC error
41
42         ; MRV11-C Pased ROM Board Definitions
43
44         177000      MR$CSR= 177000          ;Default pasins control CSR address
45         160000      MR$WIN= 160000          ;Default pase window base address
46
47         ; Miscellaneous Definitions
48
49         000010      RETRY= 8.                ;Number of retries
50         004000      STACK= 4000             ;Default stack pointer for bootstraps
51
52         .MACRO ERROR TEXT
53         HALT
54         .IRP PC$,\.
55         .SBTTL -----> HALT AT PC='PC$ INDICATES "'TEXT"
56         .ENDR
57         .ENDM ERROR

```

```

1          .SBTTL Memory Test
2
3 000000          .ASECT
4          .IIF NDF ORIGIN, ORIGIN=173000 ;Default assembly base is boot ROM area
5          .=ORIGIN ;Orisin to besinnins of bootstrap ROM
6
7          ; Start of Memory Diagnostics
8          ;
9
10 173000 012700 000340 MEM:  MOV    #340,R0          ;Assure that interrupts are disabled
11 173004 106400          MTPS   R0              ; by setting priority to PR7
12 173006 000005          RESET          ;Reset all devices (assert BINIT)
13 173010 111701          MOVBE  @PC,R1         ;Set R1=4 **** Depends on CLR R4 following!
14 173012 005004          CLR    R4              ;Set initial address for sizing
15 173014 012721 173032 MOV    #2#+173000-MEM,(R1)+ ;Set timeout trap to go to 2$
16 173020 010011          MOV    R0,@R1          ; at priority 7
17 173022 011706          MOV    @PC,SP         ;And set stack to 011414 ***** Depends on
18 173024 011414 1$:    MOV    @R4,@R4         ;Size writable memory ***** this sequence!
19 173026 005724          TST   (R4)+          ;If no trap on DATI or DATO, skip to next adrs
20 173030 000775          BR    1$            ;Loop until timeout occurs
21
22 173032 005744 2$:    TST   -(R4)          ;R4 -> highest writable memory address
23 173034 005003          CLR    R3            ;Clear initial memory test data
24
25          ; Subroutine to Write, Read, and Verify Memory
26          ; Enter with:
27          ;     R3 = 0
28          ;     R4 = High address for verification
29          ;
30          ; The contents of R2 are destroyed.
31          ; The test consists of a memory address and data storage test.
32          ; First we write all of memory with its address and then read
33          ; and verify memory.
34
35 173036 005002 MEMMO: CLR    R2              ;Copy starting address
36 173040 010212 ADDR1: MOV    R2,@R2         ;Store address at address
37 173042 005722          TST   (R2)+          ;Update to next word address
38 173044 020204          CMP    R2,R4          ;Finished with all addresses?
39 173046 101774          BLOS  ADDR1          ;If LOS no
40 173050 024202 CHECK:  CMP    -(R2),R2     ;Yes, check data
41 173052 001401          BEQ   ADCONT         ;If EQ, no comparison error
42 173054 MEMERA: ERROR <Memory address error>
43          ; Expected data is in R2; bad data is pointed to by adrs in R2.
44          ; Type 'P' to continue test.
45
46 173056 005702 ADCONT: TST   R2              ;Have we finished the check?
47 173060 001373          BNE   CHECK          ;If NE, no

```

```

1          .SBTTL Memory Data Storage Test
2
3          ; Memory data storage test.
4          ;
5          ; The steps of this test are:
6          ; 1. Fill memory with zeroes.
7          ; 2. Walk an all 1's word through memory & verify each location.
8          ; 3. Fill memory with ones.
9          ; 4. Walk an all 0's word through memory & verify each location.
10         ; By performing these steps all bit positions are checked for 0/1
11         ; storage and the sense amps are stressed in the semiconductor memories.
12
13 173062 010322 MEMT:  MOV    R3,(R2)+      ;Move background data to memory
14 173064 020204      CMP    R2,R4          ;Done with desired area?
15 173066 101775      BLOS   MEMT          ;If LOS, no
16 173070 005103 WALK:  COM    R3              ;Complement test data
17 173072 074342      XOR    R3,-(R2)       ;Load test data in memory location @(R2-2)
18 173074 005112      COM    @R2          ;Check for correct data
19 173076 001401      BEQ    DCONT        ;If EQ, data is good
20 173100      MEMERD: ERROR: <Bad memory data>
21         ; Expected data is in R3; bad data is pointed to by adrs in R2.
22         ; Type 'P' to continue test.
23
24 173102 005103 DCONT:  COM    R3              ;Get previous background data
25 173104 074312      XOR    R3,@R2       ;Restore background data for current location
26 173106 005702      TST    R2              ;Done with one pass?
27 173110 001367      BNE    WALK        ;If NE, no
28 173112 005103      COM    R3              ;Flip to other pattern
29 173114 001362      BNE    MEMT        ;If NE, test not complete yet
30 173116 010406      MOV    R4,SP          ;Set stack to top of writable memory
31         ;
32 173120      RLBOOT:

```

```

1                                     .SBTTL  RL01 Bootstrap
2
3                                     ; The following routine will bootstrap unit #0 of the RL01 disk
4                                     ; system, if a cartridge is present.
5
6 173120 004567 000574      RLBOOT: JSR      R5,SETT4      ;Set timeout trap to start
7 173124 173300             .WORD      RKBOOT-MEM+173000 ; RK05 boot if RL is not present,
8 173126 174400             .WORD      RLCS          ; And load RLCS adrs into R1
9 173130 012702 000010      MOV       #RETRY,R2      ;R2 = total retry count for all errors
10 173134 012700 174404     1$:      MOV       #RLDA,R0      ;R0 -> RLDA register
11 173140 012720 000013     MOV       #RLG#RS,(R0)+ ;Set reset and set status code in RLDA
12 173144 004567 000074     JSR      R5,3$          ;And so start
13 173150 000004             .WORD      RL$GST          ; a set status operation
14 173152 032711 000001     BIT       #RL#$DR,@R1   ;Is drive ready?
15 173156 001011             BNE      2$            ;If NE, yes
16 173160 016103 000006     MOV       6(R1),R3      ;Else set error status
17 173164 042703 177730     BIC      #RL#$ER,R3     ;Clear all but state and cover open info
18 173170 001443             BEQ      RKBOOT        ;If EQ, no cartridge is present
19 173172 022703 000006     CMP      #RLE$UH,R3    ;Unload heads, spin down, or cover open?
20 173176 101440             BLOS     RKBOOT        ;If LOS yes - so try RK05
21 173200 000755             BR       1$            ;Else wait for drive ready
22
23 173202 004514             2$:      JSR      R5,@R4          ;Start read header operation
24 173204 000010             .WORD      RL$RHD        ; to set current head position
25 173206 011003             MOV       @R0,R3        ;Get disk address info from RLMP
26 173210 042703 000177     BIC      #RL#$CA,R3     ;Clear all but cylinder address
27 173214 005203             INC      R3             ;Set for seek function
28 173216 010340             MOV      R3,-(R0)      ;Place cylinder offset in RLDA
29 173220 004514             JSR      R5,@R4        ;Go do seek operation
30 173222 000006             .WORD      RL$SEK        ; to cylinder 0, head 0
31 173224 005037 174402     CLR      @#RLBA        ;Prepare to read into location 0
32 173230 005020             CLR      (R0)+         ; from cyl 0 head 0 sector 0
33 173232 012710 177400     MOV      #-256.,@R0    ; for 256. words
34 173236 004514             JSR      R5,@R4        ;Start read operation
35 173240 000014             .WORD      RL$RD        ; for first block of data
36 173242 000413             BR       6$            ;And so check for valid boot
37
38 173244 010704             3$:      MOV      PC,R4          ;Make subsequent calls easier
39 173246 012511             MOV      (R5)+,@R1     ;Start operation on RL
40 173250 032711 100200     4$:      BIT      #100200,@R1 ;Wait for done or error
41 173254 001775             BEQ      4$            ;If EQ, neither set set
42 173256 100401             BMI     5$            ;If MI, an error occurred
43 173260 000205             RTS      R5           ;Else return
44
45 173262 005726             5$:      TST      (SF)+      ;Dump return address
46 173264 077255             SOB      R2,1$        ;And retry operation
47 173266             ERROR    <RL Boot Failure>
48 173270 000713             BR       RLBOOT       ;Proceed will restart RL bootstrap
49
50 173272 005000             6$:      CLR      R0          ;Set unit 0
51 173274 004567 000406     JSR      R5,CHK240     ;Check for valid secondary boot,
52
53 173300      RKBOOT:

```

```

1                                     .SBTTL  RK05 Bootstrap
2
3                                     ; The following routine will bootstrap the lowest-number RK05 unit which is
4                                     ; ready and operational.  If none are found, it will proceed to the RX boot.
5
6 173300 004567 000414  RKBOOT: JSR  R5,SETT4      ;Set up timeout trap to start
7 173304 173442          .WORD  RXBOOT-MEM+173000 ; if RK05 is not present, and
8 173306 177412          .WORD  RKDA          ; Preload RKDA into R1
9 173310 004567 000430  JSR  R5,DELAY4      ;If RKV11 present, delay 8 seconds for spinup
10 173314 005003         CLR  R3          ;Initialize unit number word
11 173316 012701 177412  1%:  MOV  #RKDA,R1      ;Prepare to load registers
12 173322 010311         MOV  R3,@R1        ;Set unit # and disk address
13 173324 005041         CLR  -(R1)         ;Bus address = 000000
14 173326 012741 177400  MOV  #-256,-(R1)    ;Word count = 256.
15 173332 012741 000005  MOV  #RK$RED,-(R1)   ;Start read operation
16 173336 032711 100200  2%:  BIT  #100200,@R1    ;Wait for error or done
17 173342 001775         BEQ  2%          ;If EQ, neither set yet
18 173344 100010         BPL  5%          ;If PL, operation successful
19 173346 012711 000001  MOV  #RK$CRS,@R1    ;Else reset controller
20 173352 105711         3%:  TSTB @R1        ;Wait for done
21 173354 100376         BPL  3%          ;If PL, not done yet
22 173356 062703 020000  4%:  ADD  #020000,R3    ;Bump unit selected
23 173362 103355         BCC  1%          ;And so try next unit
24 173364 000426         BR   RXBOOT       ;Else try floppies
25
26 173366 010300         5%:  MOV  R3,R0        ;Copy current unit #
27 173370 006300         ASL  R0          ;Shift unit # down
28 173372 006100         ROL  R0          ; for secondary bootstrap
29 173374 006100         ROL  R0
30 173376 006100         ROL  R0
31 173400 004567 000302  JSR  R5,CHK240     ;Check for valid secondary boot
32 173404 000764         BR   4%          ;If failure, try next unit

```



```

1          .SBTTL MRV11-C Bootstrap
2
3          ; The following routine loads the first 256 words stored on an MRV11-C
4          ; ROM board into low memory and executes them, if the first word is a
5          ; NOP (i.e., follows PDP-11 bootstrap conventions). The MRV11-C must
6          ; be strapped for paging mode enable and the correct window address.
7          ; It is the responsibility of the code loaded from the MRV11-C board to
8          ; load any additional program area from the board into RAM.
9
10 173406 004567 000306 MRBOOT: JSR    R5,SETT4      ;Setup timeout trap to start
11 173412 173120          .WORD    RLBOOT-MEM+173000 ; RL01 boot if MRV11-C is not present,
12 173414 177000          .WORD    MR$CSR          ; and preload CSR address into R1
13 173416 005011          CLR      @R1           ;Set paging CSR to map page 0
14 173420 012700 001000  MOV      #512,R0          ;Load high byte offset
15 173424 016040 157776 1$:      MOV      MR$WIN-2(R0),-(R0) ;Copy word from ROM to RAM
16 173430 005700          TST      R0           ;Have we loaded all 256 words yet?
17 173432 001374          BNE      1$           ;If NE, no
18 173434 004567 000246  JSR      R5,CHK240        ;Check for valid secondary boot
19 173440 000627          BR       RLBOOT         ;And if not valid, pass on to RL01 boot

```

```

1          .SBTTL  RX01/RX02 Bootstrap
2
3          ; This routine will bootstrap either floppy drive, at the density of the
4          ; media mounted in that drive.
5          ;
6          ; Register usage:
7          ;   R0 = density bit ! unit select bit (proto for commands)
8          ;   R1 = RXDB address
9          ;   R2 = bus address for next read operation
10         ;   R3 = word count/sector
11         ;   R4 = RXGO TR/DONE test routine pointer
12         ;   R5 = current sector address (1,3,5,7)
13
14         .ENABL  LSB
15 173442 004567 000252  RXBOOT: JSR    R5,SETT4      ;Set timeout trap to restart MR boot
16 173446 173406          .WORD  MRBOOT-MEM+173000 ; if RX is not present, and
17 173450 177172          .WORD  RXDB          ; preload RXDB into R1
18 173452 004567 000272  JSR    R5,DELAY1     ;Wait for 2 seconds if RXU is present
19 173456 005046          CLR    -(SP)          ;Set unit=0
20 173460 000404          BR     RXTRY          ;And start boot
21
22 173462 012700 000020  RXDOVER: MOV   #RX$$UN,R0      ;Get unit number mask
23 173466 074016          XOR    R0,@SP          ;And switch units
24 173470 001746          BEQ    MRBOOT          ;if both have been tried, so on to next boot
25 173472 111600          RXTRY: MOV   @SP,R0          ;Initialize current unit/density word
26 173474 004567 000154  JSR    R5,RXGO          ;Start a read status operation
27 173500 000013          .WORD  RX$RST          ; to determine status and density
28 173502 111102          MOV   @R1,R2          ;Pick up low byte of status
29 173504 100366          BPL   RXDOVER          ;If PL, drive not ready
30 173506 012703 000100  MOV    #64.,R3          ;Assume single density, set word count
31 173512 032702 000040  BIT    #RXE$DN,R2      ;Check media density
32 173516 001403          BEQ    1$              ;If EQ, single density
33 173520 052700 000400  BIS    #RX$$DE,R0      ;Else set double density in default word
34 173524 006303          ASL   R3              ;And double word count/sector
35 173526 005002          1$: CLR   R2          ;Current bus address = 000000
36 173530 012705 000001  MOV    #1,R5          ;Current sector address = 1
37 173534 004567 000114  2$: JSR   R5,RXGO          ;Start read sector operation
38 173540 000007          .WORD  RX$RED          ; and wait for TR
39 173542 010511          MOV   R5,@R1          ;Set sector #
40 173544 004514          JSR   R5,@R4          ;Wait for TR again
41 173546 012711 000001  MOV    #1,@R1          ;Set track #1
42 173552 004514          JSR   R5,@R4          ;Wait for DONE
43 173554 100714          BMI   MRBOOT          ;If error, skip this boot
44 173556 004567 000072  JSR   R5,RXGO          ;Start empty buffer function
45 173562 000003          .WORD  RX$EMP          ; and wait for TR
46 173564 032737 004000 177170  BIT    #RX$$02,@#RXCS  ;Is DMA available?
47 173572 001413          BEQ    3$              ;If EQ no - handle as RX01
48 173574 010311          MOV   R3,@R1          ;Else load word count
49 173576 004514          JSR   R5,@R4          ;Wait for TR
50 173600 010211          MOV   R2,@R1          ;And load current bus address
51 173602 004514          JSR   R5,@R4          ;Wait for DONE
52 173604 122525          CMPB  (R5)+,(R5)+     ;Update sector adrs
53 173606 060302          ADD   R3,R2          ;And also the
54 173610 060302          ADD   R3,R2          ; current bus address
55 173612 022702 001000  CMP    #512.,R2        ;Have we read all of first block yet?
56 173616 001346          BNE   2$              ;If NE, no - continue with next sector
57 173620 000404          BR    5$              ;Else so check for valid secondary boot

```

```

58
59 173622 006303          3$:   ASL      R3           ;Turn word count into byte count
60 173624 111122          4$:   MOVEB   @R1,(R2)+      ;Move one byte from buffer to memory
61 173626 004514          JSR      R5,@R4          ;Wait for TR or DONE
62 173630 077303          SOB      R3,4$          ;Loop for all bytes in first sector
63 173632 005000          5$:   CLR      R0           ;Assume unit #0
64 173634 105716          TSTB   @SF             ;Was it unit #0?
65 173636 001401          BEQ     6$             ;If EQ, yes
66 173640 005200          INC     R0           ;Else indicate unit #1
67 173642 005741          6$:   TST     -(R1)        ;Reset to point to CSR
68 173644 004567 000036 JSR     R5,CHK240      ;Check for valid secondary boot
69 173650 005721          TST     (R1)+         ;If not valid, reset to RXDB
70 173652 000703          BR     RXOVER         ;And so switch units
71          .DSABL  LSB
72
73 173654 012504          RXG0:  MOV     (R5)+,R4      ;Copy command word to use
74 173656 050004          BIS     R0,R4         ;Set unit # and density
75 173660 010437 177170 MOV     R4,@#RXCS      ;Start operation
76 173664 010704          MOV     PC,R4         ;Copy adrs for later calls
77 173666 005741          TST     -(R1)        ;R1 -> RXCS
78 173670 032711 000240 1$:   BIT     #RX$$TR!RX$$DN,@R1 ;Wait for TR or DONE
79 173674 001775          BEQ     1$           ;If EQ, neither are true yet
80 173676 005721          TST     (R1)+         ;Reset R1 -> RXDB and check for errors
81 173700 000205          RTS     R5           ;Return to caller

```

```

1                                     .SBTTL  Miscellaneous Subroutines
2
3 173702  005737  170000          CRUNCH: TST      @#170000          ;This better trap to 4, guys
4
5                                     ; The CHK240 routine will return only if location 0 does not contain a
6                                     ; valid secondary bootstrap (i.e., does not have a NOP instruction in it),
7                                     ; otherwise it starts execution of the booted program.
8
9 173706  022737  000240  000000  CHK240: CMP      #240,@#0          ;Did we read a valid bootstrap?
10 173714  001024          BNE      RETR5          ;If NE, no
11 173716  005007          CLR      PC          ;Else so to it
12
13                                     ; The SETT4 routine is used to set up the bus timeout trap vector to
14                                     ; automatically activate another bootstrap if the current device is not
15                                     ; present on the system.
16
17 173720  012537  000004          SETT4: MOV      (R5)+,@#4          ;Set trap adrs into location 4
18 173724  012737  000340  000006  MOV      #340,@#6          ;And PS = Priority 7
19 173732  012501          MOV      (R5)+,R1          ;Preload R1
20 173734  012706  004000          MOV      #STACK,SP          ;Reset stack pointer
21 173740  005711          TST      @R1          ;Trap to next boot if controller not present
22 173742  000115          JMP      @R5          ;And return to caller
23
24                                     ; The DELAY routines are used by the RK05 and RX bootstraps to wait for
25                                     ; the device to spin up. Bus timeout traps (min. 10us) are used to minimize
26                                     ; processor speed dependence.
27
28 173744  004517          DELAY4: JSR      R5,@PC          ;Wait for min. 7 seconds
29 173746  004517          JSR      R5,@PC
30 173750  012700  000004          DELAY1: MOV      #4,R0          ;Wait for min 1.79 seconds
31 173754  010604          MOV      SP,R4          ;Save current stack pointer
32 173756  005003          CLR      R3          ;Set maximum count for loop
33 173760  010710          MOV      PC,@R0          ;Set timeout PC (Assumes @#6 = 340!)
34 173762  010406          MOV      R4,SP          ;Restore original SP
35 173764  077332          SOB      R3,CRUNCH          ;If R3 NE 0, so force timeout trap
36 173766  000205          RETR5: RTS      R5          ;Handy RTS R5 instruction

```

UNIVERSAL DISK BOOTSTRAP  
MISCELLANEOUS SUBROUTINES

MACRO V03.02B13-JUN-79 PAGE 11

21  
26           000001

; YOU HAVE 4 FREE WORDS UNUSED OUT OF 256.  
      .END

UNIVERSAL DISK BOOTSTRAP  
SYMBOL TABLE

MACRO V03.02B13-JUN-79 PAGE 11-1

ADCONT	173056	RETRS	173766	RLBOOT	173120	RXCS	= 177170	RX\$WDD=	000015
ADDRT	173040	RKBA	= 177410	RLCS	= 174400	RXDB	= 177172	RX\$WRT=	000005
CHECK	173050	RKBOOT	173300	RLDA	= 174404	RXE\$CR=	000001	RX\$\$DE=	000400
CHK240	173706	RKCS	= 177404	RLE\$UH=	000006	RXE\$DD=	000100	RX\$\$DN=	000040
CRUNCH	173702	RKDA	= 177412	RLG\$RS=	000013	RXE\$DE=	000020	RX\$\$ER=	100000
DCONT	173102	RKDB	= 177416	RLMP	= 174406	RXE\$DN=	000040	RX\$\$FN=	000016
DELAY1	173750	RKDS	= 177400	RL\$GST=	000004	RXE\$DR=	000200	RX\$\$GO=	000001
DELAY4	173744	RKER	= 177402	RL\$NOP=	000000	RXE\$ID=	000004	RX\$\$IE=	000100
MEM	173000	RKWC	= 177406	RL\$RD	= 000014	RXE\$UN=	000400	RX\$\$IN=	040000
MEMIO	173036	RK\$CRS=	000001	RL\$RDN=	000016	RXGO	173654	RX\$\$TR=	000200
MEMERA	173054	RK\$DRS=	000015	RL\$RHD=	000010	RXOVER	173462	RX\$\$UN=	000020
MEMERD	173100	RK\$RCK=	000013	RL\$SEK=	000006	RXTRY	173472	RX\$\$XA=	030000
MEMT	173062	RK\$RED=	000005	RL\$WCK=	000002	RX\$EMP=	000003	RX\$\$XX=	003000
MRBOOT	173406	RK\$SEK=	000011	RL\$WD	= 000012	RX\$FIL=	000001	RX\$\$02=	004000
MR\$CSR=	177000	RK\$WCK=	000007	RL\$\$CA=	000177	RX\$REC=	000017	SETT4	173720
MR\$WIN=	160000	RK\$WLK=	000017	RL\$\$DR=	000001	RX\$RED=	000007	STACK	= 004000
ORIGIN=	173000	RK\$WRT=	000003	RL\$\$ER=	177730	RX\$RST=	000013	WALK	173070
RETRY	= 000010	RLBA	= 174402	RXBOOT	173442	RX\$STD=	000011		

. ABS. 173770 000  
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 1624 WORDS ( 7 PAGES)  
DYNAMIC MEMORY AVAILABLE FOR 48 PAGES  
;LP:DISKBT=DK:DISKBT


<b>μnote</b>		NUMBER 063
TITLE	RL01 Type-In Bootstrap	DATE 3 / 19 / 79
DISTRIBUTION	Unrestricted	PRODUCT RL01
ORIGINATOR	Barbara Beck	PAGE 1 OF 1

The following bootstrap may be typed in, under ODT control, to boot an RL01 disk, drive 0. To boot other than drive 0, locations 1020, 1036 and 1062 should contain the drive number in the upper byte.

RL01 TYPE-IN BOOT

1000	012710	MOV	#174400,R0	;Put CSR in R0
1002	174400			
1004	105710	TSTB	(R0)	;Check for Ready
1006	100376	BPL	.-2	;Wait for Controller
1010	012760	MOV	#13,4(R0)	;Set Up for Drive
1012	13			;Reset and Clear Error
1014	4			
1016	012710	MOV	#4,(R0)	;Get Status
1020	4			
1022	105710	TSTB	(R0)	;Wait for Ready
1024	100376	BPL	.-2	;Loop on Wait
1026	012760	MOV	#77601,4(R0)	;Seek - Cyl
1030	077601			
1032	4			
1034	012710	MOV	#6,(R0)	;Seek Command
1036	6			
1040	105710	TSTB	(R0)	;Check for Ready
1042	100376	BPL	.-2	;Loop on Wait
1044	012760	MOV	#177400,6(R0)	;Word Count-400
1046	177400			;(2 Sectors)
1050	6			
1052	012760	MOV	#0,4(R0)	;Clear Disk Address
1054	0			;Register
1056	4			
1060	012710	MOV	#14,(R0)	;Read
1062	14			
1064	105710	TSTB	(R0)	;Check for Ready
1066	100376	BPL	.-2	;Loop on Wait
1070	005710	TST	(R0)	;Check for Error
1072	100001	BPL	.+2	;Branch if OK
1074	000000	HALT		;Halt on Error
1076	005007	CLR	R7	;Execute Boot

**digital**  
**COMPONENTS**  
**GROUP**

 TITLE <u>DLV11-J I/O Page Address Problem Report</u> DISTRIBUTION <u>DLV11-J Users</u> ORIGINATOR <u>Joe Austin</u>	NUMBER 064
	DATE 4 / 03 / 79
	PRODUCT DLV11-J
	PAGE 1 OF 2

PROBLEM

Under certain conditions, the DLV11-J will falsely respond to bus cycles intended for other bus interface modules. When this happens, the DLV11-J address selection logic will be enabled, in addition to the address selection logic of the correct module. This results in the DLV11-J placing information onto the bus which will be OR'ed with the data from the correct module.

NOTE: DLV11-J modules at Circuit Schematic (CS) Revision E or above do not have this problem.

CONDITIONS

- 1) This problem will occur only when the program is performing a bus cycle that accesses the I/O page and will not occur for accesses in the 0-28K word address space for the LSI-11.
- 2) This problem will occur only on DLV11-J modules that are configured to have a console port.
- 3) This problem is more noticeable with the 11/23 and with DMA transfers on the LSI-11 and LSI-11/2.

PROBLEMS OBSERVED

- 1) This problem has been noted to occur when using the bootstrap in the REV11 module to load paper tape. In this case, the REV11 ROM command "AL177560" will cause an abort to ODT. However, the command "AL CR" will work properly.
- 2) Program failures and data errors may be noted when using the 28K to 30K area in the I/O page for program memory.



- 3) Errors may also be noted while using other I/O interfaces which have certain address bits (notably bits 5 and 8-12) that are similar to those used by the DLV11-J.

SOLUTION

Install ECO M8043-002 on the DLV11-J. This ECO brings the CS up to Revision E.

All modules processed by the Customer Repair Area (CRA) will be updated to this ECO. This update will be performed at no charge to the customer. This customer should obtain a no-charge SBA from his Sales Specialist in order to have the module updated.

QUICK CHECK

The presence of green wires or of CS Revision E or higher indicates that this ECO has been installed.

NOTE

THIS ECO IS REQUIRED FOR ALL DLV11-J  
MODULES USED WITH THE LSI-11/23.

<h1>μnote</h1>	NUMBER	065
	DATE	6 / 04 / 79
	PRODUCT	RX02
	PAGE 1 OF 3	
TITLE	Bootstrap for RX02	
DISTRIBUTION	Unrestricted	
ORIGINATOR	Barry Maskas	

These boots are to be used in systems with RX02 floppy disk drives controlled by a Q-bus (M8029) interface. The diskette for boot #1 must be DIGITAL double density and must reside in Drive 0. The diskette for boot #2 must be single density format and must reside in Drive 0.

The LTC must be off and other interrupts inhibited; i.e., under ODT enter \$S/-----340(CR). Initialize the stack pointer: \$6/-----1000(CR). Initialize the program counter: \$7/-----1000(CR). Type in the appropriate boot, return to address 1000 and type P; i.e., 1000P

RX01 TYPE-IN BOOT #1 -- DOUBLE DENSITY DISKETTE FORMAT

```

001000 012700 BEGIN: MOV #100240,R0 ;INIT TEST WORD
001002 100240
001004 012701 MOV #177170,R1 ;INIT RX2CS ADDR
001006 177170
001010 005002 CLR R2 ;INIT BUS ADDR
001012 012705 MOV #200,R5 ;INIT WDCNT
001014 000200
001016 012704 MOV #401,R4 ;INIT TRACK & SECTOR ADDR
001020 000401
001022 012703 START: MOV #177172,R3 ;INIT RX2DB ADDR
001024 177172
001026 030011 WAIT: BIT R0,(R1) ;WAIT FOR READY,
001030 001776 BEQ WAIT ;DONE AND ERROR FLAGS
001032 100437 BMI HALT ;HALT ON ERROR
001034 012711 MOV #407,(R1) ;READ SECTOR
001036 000407
001040 030011 WAIT1: BIT R0,(R1) ;WAIT FOR READY
001042 001776 BEQ WAIT1
001044 100432 BMI HALT ;HALT ON ERROR
001046 110413 MOV R4,(R3) ;LOAD SECTOR ADDR
001050 000304 SWAB R4 ;INIT TRACK ADDR
001052 030011 WAIT2: BIT R0,(R1) ;WAIT FOR READY
001054 001776 BEQ WAIT2
001056 110413 MOV R4,(R3) ;LOAD TRACK ADDR
001060 000304 SWAB R4 ;REINIT SECTOR ADDR

```

**digital**  
**COMPONENTS**  
**GROUP**

```

001062 030011 WAIT3: BIT R0, (R1) ;WAIT FOR DONE
001064 001776 BEQ WAIT3
001066 100421 BMI HALT ;HALT ON ERROR
001070 012711 MOV #403, (R1) ;EMPTY BUFFER
001072 000403
001074 030011 WAIT4: BIT R0, (R1) ;WAIT FOR READY
001076 001776 BEQ WAIT4
001100 100414 BMI HALT ;HALT ON ERROR
001102 010513 MOV R5, (R3) ;LOAD WORD CNT
001104 030011 WAIT5: BIT R0, (R1) ;WAIT FOR READY
001106 001776 BEQ WAIT5
001110 100410 BMI HALT ;HALT ON ERROR
001112 010213 MOV R2, (R3) ;LOAD BUS ADDR
001114 060502 ADD R5, R2 ;UPDATE BUS ADDR
001116 060502 ADD R5, R2
001120 122424 CMPB (R4)+, (R4)+ ;UPDATE SECTOR ADDR
001122 120427 CMPB R4, #3 ;READ ONE BLOCK?
001124 000003
001126 003735 BLE START ;IF NO, AGAIN
001130 005007 CLR PC ;IF YES, LOAD
001132 000000 HALT: HALT

```

RX02 TYPE-IN BOOT #2 -- SINGLE DENSITY DISKETTE FORMAT

```

001000 012700 BEGIN: MOV #100240, R0 ;INIT TEST WORD
001002 100240
001004 012701 MOV #177170, R1 ;INIT RX2CS ADDR
001006 177170
001010 005002 CLR R2 ;INIT BUS ADDR
001012 012705 MOV #100, R5 ;INIT WDCNT
001014 000100
001016 012704 MOV #401, R4 ;INIT TRACK AND SECTOR
001020 000401
001022 012703 START: MOV #177172, R3 ;INIT RX2DB ADDR
001024 177172
001026 030011 WAIT: BIT R0, (R1) ;WAIT FOR READY,
001030 001776 BEQ WAIT ;DONE AND ERROR
001032 100437 BMI HALT ;HALT ON ERROR
001034 012711 MOV #007, (R1) ;READ SECTOR
001036 000007
001040 030011 WAIT1: BIT R0, (R1) ;WAIT FOR READY
001042 001776 BEQ WAIT1
001044 100432 BMI HALT ;HALT ON ERROR
001046 110413 MOV R4, (R3) ;LOAD SECTOR ADDR
001050 000304 SWAB R4 ;INIT TRACK ADDR
001052 030011 WAIT2: BIT R0, (R1) ;WAIT FOR READY
001054 001776 BEQ WAIT2
001056 110413 MOV R4, (R3) ;LOAD TRACK ADDR
001060 000304 SWAB R4 ;REINIT SECTOR ADDR

```

```
001062 030011 WAIT3: BIT R0, (R1) ;WAIT FOR DONE
001064 001776 BEQ WAIT3
001066 100421 BMI HALT ;HALT ON ERROR
001070 012711 MOV #003, (R1) ;EMPTY BUFFER
001072 000003
001074 030011 WAIT4: BIT R0, (R1) ;WAIT FOR READY
001076 001776 BEQ WAIT4
001100 100414 BMI HALT ;HALT ON ERROR
001102 010513 MOV R5, (R3) ;LOAD WORD CNT
001104 030011 WAIT5: BIT R0, (R1) ;WAIT FOR READY
001106 001776 BEQ WAIT5
001110 100410 BMI HALT ;HALT ON ERROR
001112 010213 MOV R2, (R3) ;LOAD BUS ADDR
001114 060502 ADD R5, R2 ;UPDATE BUS ADDR
001116 060502 ADD R5, R2
001120 122424 CMPB (R4)+, (R4)+ ;UPDATE SECTOR
001122 120427 CMPB R4, #7 ;READ ONE BLOCK?
001124 000007
001126 003735 BLE START ;IF NO, AGAIN
001130 005007 CLR PC ;IF YES, LOAD
001132 000000 HALT: HALT
```



TITLE 11/23 Floating Point Compatibility  
DISTRIBUTION Restricted to KEF11-AA Users  
ORIGINATOR Barry Maskas

NUMBER	066
DATE	6 / 08 / 79
PRODUCT	KEF11-AA
PAGE 1 OF 1	

Early KDF11 (M8186) modules will not support floating point functionality because the FP11 Hybrid I.C. is not compatible with initial versions of the CTL-DAT Hybrid chip and MMU chip. These modules can be identified by the following or earlier revision codes or part numbers printed on the chips:


	<u>COMBINATION</u>	<u>MODEL #</u>	<u>CHIP PN</u>	<u>HYBRID PN</u>
CTL:	#1	DEC 303-C	23-001C7-AA)	57-00000-00
DAT:		DEC 302-E	21-15541-AA)	
MMU:		DEC 304-C	21-15542-00	
CTL:	#2	DEC 303-D	23-001C7-AA)	57-00000-00
DAT:		DEC 302-E	21-15541-AA)	
MMU:		DEC 304-C	21-15542-00	
CTL:	#3	DEC 303-D	23-001C7-AA)	57-00000-01
DAT:		DEC 302-F	21-15541-AB)	
MMU:		DEC 304-C	21-15542-00	

All later revision codes are compatible with the option.

If a user (DCG customers only) desires to upgrade his LSI-11/23 module to include floating point, he should contact his microcomputer sales representative.

NOTE: With the exception of floating point, the modules with the above chip sets have identical functionality to the upgraded modules.

**digital**  
**COMPONENTS**  
**GROUP**

	NUMBER 067A
	DATE 8 / 16 / 79
	PRODUCT DLV11-J
	PAGE 1 OF 2

THIS MICRO NOTE SUPERSEDES MICRO NOTE #067

This Micro Note describes the fix to a potential problem that has been reported by some DLV11-J users.

SYMPTOMS

- 1) The system can transmit data to a terminal connected to the DLV11-J but cannot receive data from it.
- 2) The output on pins 6 and 7 of the 9637 ATC receiver chip (E34 or E37) is constantly low while data is observed on the inputs (pins 2 or 3) of the receiver chip.

PROBLEM

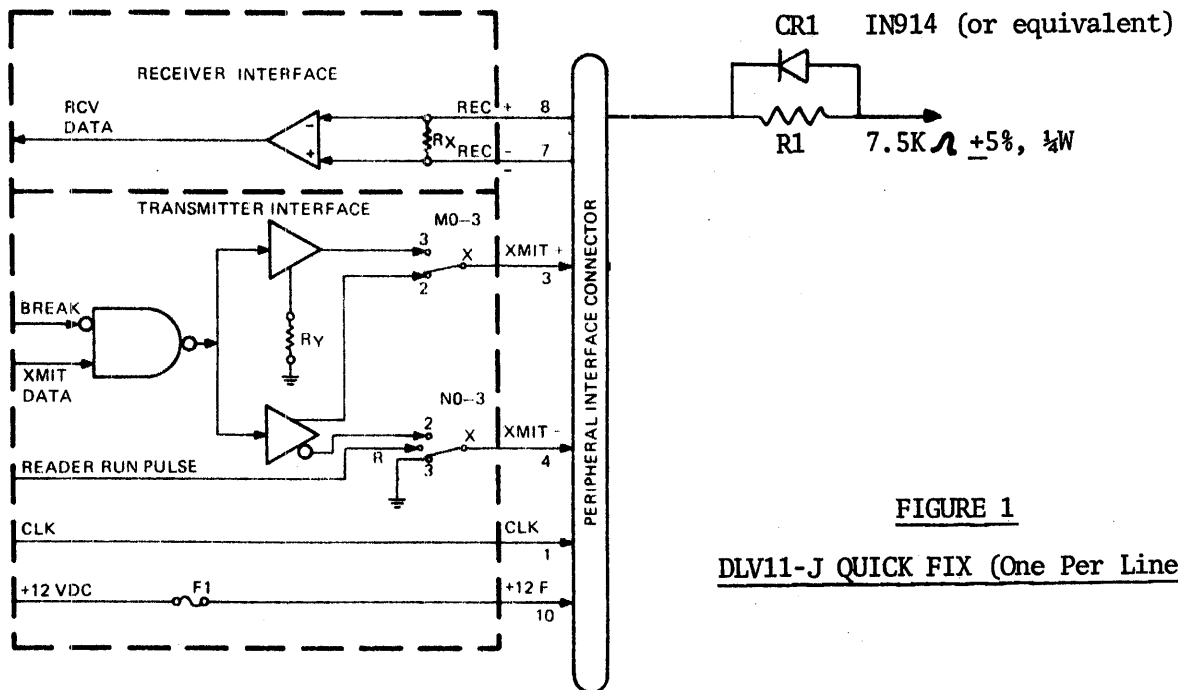
Receiver chips (2 per module) with part number 9637 ATC do not meet the original DEC specifications. If the REC+ signals on pin 8 of the cable connector (see Figure 1) becomes more negative than -10 volts, then no signal will pass through the receiver. In this case, the receiver output usually remains low.

CHIP IDENTIFICATION

All 9637 parts with a date code "79xx" or higher are correct and will work properly.

SOLUTION

A "quick fix" is described on page 2. The long-term fix consists of having the receiver chips replaced at no cost to the customer.




**FIGURE 1**  
**DLV11-J QUICK FIX (One Per Line)**

QUICK FIX

If your system experiences this problem, then the addition of the series resistor and diode as shown in Figure 1 will adequately solve it. The purpose of the resistor (R1) is to offset the input signal from the terminal such that the signal into the receiver does not exceed the operating limits of the chip. The purpose of the diode (CR1) is to prevent the positive portion of the input signal from being offset by the resistor.

NOTE: It is recommended that this fix not be used unless necessary.

	NUMBER 068
	DATE 7 / 03 / 79
	PRODUCT A11 LSI-11 Bus Modules
	PAGE 1 OF 1

The following information defines the environment in which LSI-11 bus modules are designed to operate:

OPERATING

Temperature: 5°C to 60°C -- Derate the maximum temperature by one degree Celcius for each 1000 feet of altitude above 8000 feet.

Relative Humidity: 10% to 90%, non-condensing

Altitude: Up to 50,000 feet (Note temperature derating above 8000 feet.)

Airflow: Sufficient air flow must be provided to limit the temperature rise across the module to 5°C for an inlet temperature of 60°C. For inlet air temperature below 55°C, air flow must be provided to limit temperature rise across the module to 10°C.

The actual air flow and fan requirements will vary greatly with mechanical design and the choice of modules, but in any event, the above requirements must be met for all modules in the system.

In addition to the operating considerations, the modules may be stored over a wider range of temperatures. When stored outside the operating range, modules should be allowed to stabilize in the operating range for a minimum of 5 minutes before operating. The storage limits are listed below for reference:

STORAGE

Temperature: -40°C to 66°C

Relative Humidity: 10% to 90%, non-condensing


Altitude: Up to 50,000 feet

The above information applies only to LSI-11 bus modules except the MMV11-A whose operating temperature is 5°C to 50°C. For external peripheral devices such as mass storage devices and terminals, the individual User's Guides should be consulted for environmental considerations.

NOTE: These are the design limits for the modules. Lower temperature limits will serve to increase the life of the product.





	NUMBER	069
	DATE	6 / 21 / 79
	PRODUCT	DCK11-AB, -AD
	PAGE 1 OF 2	
TITLE	18-Bit DMA With Chipkits	
DISTRIBUTION	Unrestricted	
ORIGINATOR	Rick Plummer	

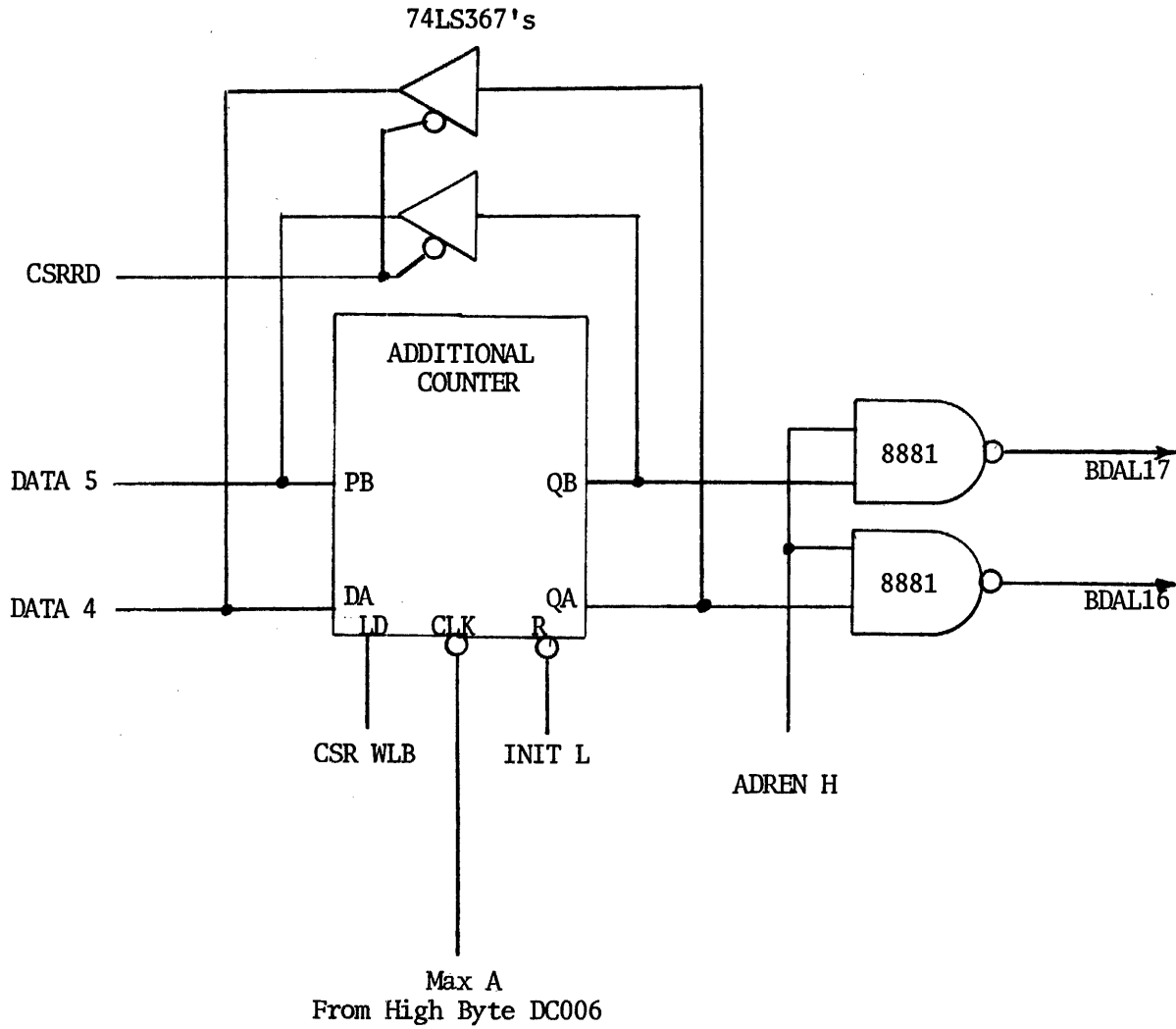
The following information will serve to assist those DMA Chipkit users who are designing interfaces for use with 18-bit addressing in LSI-11/23 systems. It is intended as an addendum to the specification and application information for the chip sets as published in the following:

- Microcomputer Memories and Peripherals Handbook
- Computer Interfacing Accessories and Logic Handbook
- Chipkit Users Manual


To extend the DMA addressing to 18 bits, the following functions must be accomplished:

1. Provide an extension of 2 bits to the bus address counter which will advance on overflow from the 16 bits provided in the DC006 chips.
2. Provide for read/write of these 2 bits as bits 4 and 5 of the interface control and status register (CSR).
3. Provide for placing these 2 bits on address lines 16 and 17 during the address portion of the DMA transfer cycle.
4. Clear the additional bits upon initialization of the bus.

The following figure outlines the required additional hardware. Signal mnemonics are from the DMA application section contained in the above-mentioned documents. Because bits 4 and 5 are used for software compatibility with other DMA interfaces, it will be necessary to use a bit other than 5 for data in/data out selection as was the case in the application information.



IMA CHIPKIT ADDRESS EXTENSION


	NUMBER	070
	DATE	6 / 22 / 79
	PRODUCT	KDF11-A
	PAGE 1 OF 2	
TITLE	LSI-11 vs. LSI-11/23 Bus Transaction Differences	
DISTRIBUTION	Unrestricted	
ORIGINATOR	Rick Plummer	

There are a number of bus transactions that are performed differently on the LSI-11/23 than on the LSI-11 and some which are unique to the LSI-11/23. While these do not affect the operation of DIGITAL or other properly-designed custom interfaces, they are presented here for the purpose of helping users more thoroughly understand the bus operations.

1. MMU Address Relocation -- When the MMU is enabled, the virtual address occurs on the bus at the beginning of each bus cycle. This is then followed by the relocated address. Once the relocated address has met the bus address set-up time requirements, then sync is issued. Note that it is the relocated address that is part of the bus timing and protocol.
2. MMU Register Transfers -- When the various MMU registers are addressed from the program, the transaction will appear on the bus as any normal CPU to I/O communication (i.e., address, data, sync, DIN/DOUT and reply will occur). It is not possible, however, to communicate with these registers from another bus master on the bus.
3. PSW Transfers -- When the PSW is explicitly addressed (MOV #340, @#177776), the transfer will appear on the bus in a similar fashion to the MMU registers except that there will be no reply. Likewise, there can be no communication from a bus master to the PSW.
4. DATOB Cycles -- On previous LSI-11 processors the byte data has been present on both bytes during a DATOB cycle or the output portion of a DATIOB cycle. The LSI-11/23 presents valid data only on the byte actually being transferred during these cycles. The data on the opposite byte is not meaningful during these transactions.
5. DATIO Operand Fetches -- On previous LSI-11 processors, the source operand for MIPS and EIS instructions was fetched using the DATIO bus cycle. The LSI-11/23 fetches these using DATI bus cycles.
6. MOVB Output Cycle -- Previous LSI-11 processors performed a DATIOB bus cycle as the last cycle of instruction execution. The LSI-11/23 performs a DATOB as the last bus cycle of instruction execution.

7. CLR and CLRB Cycles -- Previous LSI-11 processors performed a DATIO (or DATIOB) cycle for hardware minimization. The LSI-11/23 uses a DATO (or DATOB) bus cycle to perform the instruction.
8. SXT Cycles -- Previous LSI-11 processors performed a DATIO cycle for the last bus cycle for hardware minimization. The LSI-11/23 uses a DATO cycle as the last bus cycle in execution.

Refer also to Micro Note 055 for bus timing differences between the LSI-11 and LSI-11/23.

	NUMBER	071
	DATE	7 / 05 / 79
	PRODUCT	BA11-M and BA11-N Boxes
	PAGE	1 OF 3
TITLE	Expanding BA11-MA and BA11-NC Based Systems	
DISTRIBUTION	Unrestricted	
ORIGINATOR	Barry Maskas	

This is intended to provide a recommended procedure for expanding BA11-MA based systems to a two or three box system which includes the BA11-NE expansion box or for expanding BA11-NC based systems to a two or three box system which includes the BA11-ME expansion box.

When expanding system configurations, pay strict attention to the single backplane configuration rules and multiple backplane configuration rules located on page 4-39 of the 1978-79 Memories and Peripherals Handbook.

The PDP-11V03 systems mounted in a H984 series cabinet (described on page 4-41 of the 1978-79 Memories and Peripherals Handbook) have only enough expansion space to house one BA11-ME expansion box.

Expansion Parts List:

Expansion Boxes:

<u>Model</u>	<u>Primary Power/Front Panel</u>	<u>Bused Slots</u>
BA11-ME	115v/blank panel	A,B,C, and D
BA11-NE	115v/blank panel	A and B only

Power Controller:

861-C	90-130v AC single phase 24A per pole primary (available from Accessories and Supplies Group at 800-258-1710) to output 90-130v AC at 12A for each outlet
-------	--

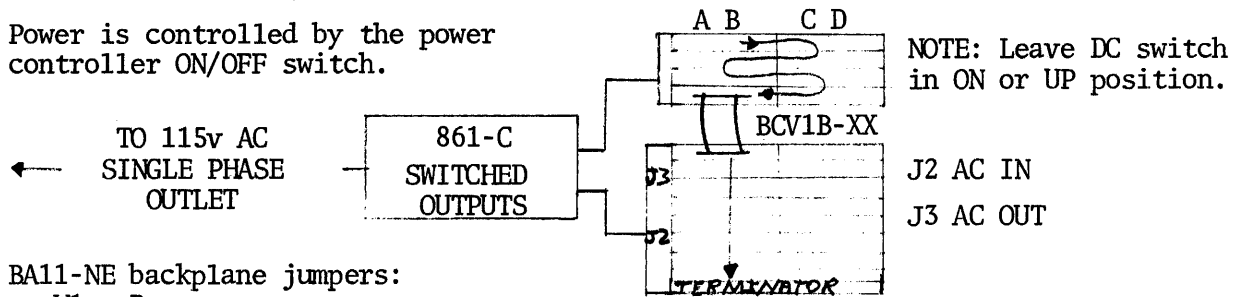
Bus Expansion Cards and Cables:

BCV1B-XX	Used with two backplane systems
BCV1A-XX	Used with the third backplane expansion

XX can be 2,4,6, or 12 foot lengths

1. BA11-MA base box expanded to a BA11-NE box:

Power is controlled by the power controller ON/OFF switch.



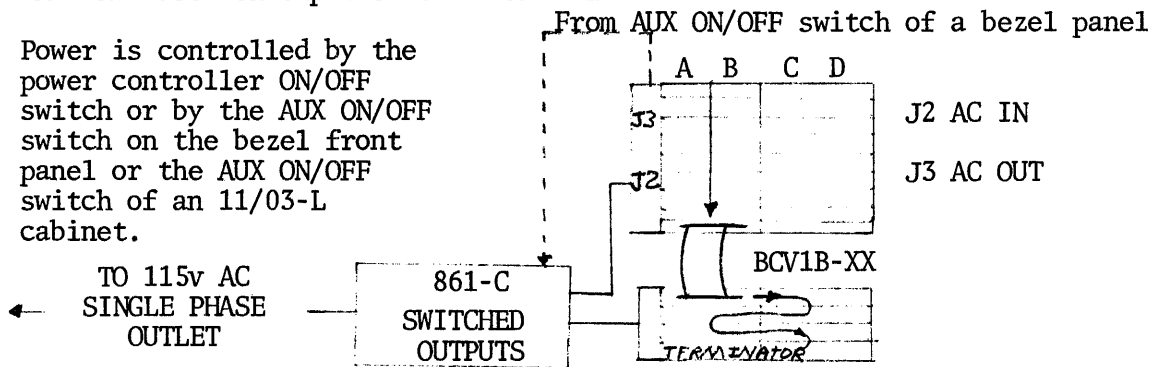
BA11-NE backplane jumpers:

- W1 R
- W2 R
- W3 R

The LTC is sourcing the BEVNT L in the BA11-MA box.

2. BA11-NC base box expanded to a BA11-ME box:

Power is controlled by the power controller ON/OFF switch or by the AUX ON/OFF switch on the bezel front panel or the AUX ON/OFF switch of an 11/03-L cabinet.



BA11-NC backplane jumpers:

- W1 I
- W2 I) If M7264 or M7264-YA CPU
- W3 I) is used, otherwise R.

BA11-NC bezel jumpers:

- W1 R) When bezel AUX ON/OFF switch is used to turn
- W2 R) system power controller on and off, otherwise I.
- W3 R
- W4 I

The LTC is sourcing the BEVNT L in the BA11-NC box via backplane jumper W1 I.

3. BA11-NC base box expanded to a BA11-NE box:

Power is controlled by the AC input box ON/OFF switch.

← TO 115v AC SINGLE PHASE OUTLET

BA11-NC backplane jumpers:

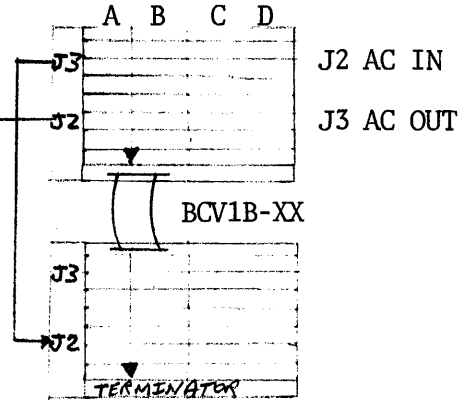
- W1 I
- W2 I) If M7264 or M7264-YA
- W3 I) CPU is used, otherwise R.

BA11-NC bezel jumpers:

- W1 I
- W2 I
- W3 R
- W4 I

BA11-NE backplane jumpers:

- W1 R
- W2 R
- W3 R




Total input current from the AC source must be less than 12A.

NOTE: Do not attempt to source AC power for the BA11-M box from the BA11-N box AC outlet since the current rating of the BA11-N box will be exceeded and severe damage may occur. Also, do not attempt to chain AC power to three BA11-N boxes.

Further expansion guidelines are contained in the BA11-N Mounting Box User's Guide (EK-BA11N-UG-001) or the BA11-N Mounting Box Technical Manual (EK-BA11N-TM-001).

The LTC is sourcing the BEVNT L in the BA11-NC box via backplane jumper W1 I.

	NUMBER 072
	DATE 7 / 12 / 79
	PRODUCT LSI-11 Memories & Peripherals
	PAGE 1 OF 2
TITLE <u>Peripheral Compatibility with 11/23 Systems</u>	
DISTRIBUTION <u>Unrestricted</u>	
ORIGINATOR <u>Barry Maskas</u>	

When configuring 11/23 systems or upgrading existing LSI-11 systems, consider the following option compatibilities:

- I. The following list of options will function in system configurations with the KDF11-AA CPU:
1. AAV11 (A6001) 4-channel 12-bit D/A converter
  2. ADV11 (A012) 16-channel 12-bit A/D converter
  3. BDV11 (M8012) bootstrap, diagnostic and terminator
  4. DLV11 (M7940) asynchronous serial line interface
  5. DLV11-E (M8017) asynchronous serial line interface (with modem controls)
  6. DLV11-F (M8028) asynchronous serial line interface
  7. DRV11 (M7941) parallel line unit
  8. DRV11-B (M7950) DMA interface
  9. DUV11 (M7951) buffered line interface
  10. DZV11 (M7957) asynchronous multiplexer
  11. IBV11-A (M7954) IEEE instrument bus interface
  12. KPV11-A, -B, -C (M8016, M8016-YB, -YC) power fail, LTC and terminator
  13. K WV11-A (M7952) programmable real-time clock
  14. LAV11 (M7949) LA180 printer interface
  15. LPV11 (M8027) LA180/LP05 printer interface
  16. MRV11-BA (M8021) UV PROM-RAM module
  17. MRV11-C (M8048) EPROM, PROM and ROM module
  18. MSV11-C (M7955-Y) MOS memory (self-refresh)
  19. MSV11-D (M8044) MOS memory (self-refresh)
  20. MXV11-AA, -AC (M8047-AA, -CA) memory and serial I/O
  21. RLV11 (M8013, M8014) RL01 disk drive controller
  22. RXV11 (M7946) RX01 floppy disk controller
  23. RXV21 (M8029) RX02 floppy disk controller
  24. TEV11 (M9400-YB) 120 ohm terminator

II. The following options can be utilized in 11/23 systems under the following conditions:

1. DLV11-J (M8043) asynchronous 4-line serial interface  
CS Revision E or higher must be used or  
ECO #M8043-MR002 must be installed.

**digital**  
**COMPONENTS**  
**GROUP**



2. MSV11-E (M8045) MOS memory with parity (self-refresh)

Requires external parity controller if parity is to be used -- see Micro Note 052.

III. The following options are incompatible in some or all 11/23 system configurations:

1. MMV11-A (H223,G653) 8KB core memory

Must be used in backplanes which supply Q-bus signals on both A/B and C/D connectors (H9270 and DDV11-B) -- it can only be configured in 64KB or less systems since it can only decode 16 bits of address.

2. MRV11-AA (M7942) PROM/ROM

Can only be configured in 64KB or less systems because it only decodes 16 bits of address.

3. MSV11-B (M7944) MOS memory (requires refresh)

Can only be configured in 64KB or less systems because it only decodes 16 bits of address and it requires external refresh.

4. REV11-A,-C (M9400-YA,-YC) bootstrap, terminator and DMA refresh

The bootstrap and diagnostics in ROM are LSI-11 unique and, therefore, use on LSI-11/23 systems is incompatible -- termination (120 ohms) and DMA refresh are features that still may be utilized.

5. RKV11-D RK05 disk interface

Can only be configured in 64KB or less systems because it only does DMA transfers into 16 bits of address.

6. KEV11 EIS/FIS chip

Designed for LSI-11 and LSI-11/2 processors and is not electrically compatible with 11/23 processors.

7. KUV11 Writable Control Store

Designed for LSI-11 and LSI-11/2 processors and is not electrically compatible with 11/23 processors.

<b>μnote</b>		NUMBER 073
TITLE	TU58 Cabling	DATE 7 / 05 / 79
DISTRIBUTION	Unrestricted	PRODUCT TU58
ORIGINATOR	Barbara Beck	PAGE 1 OF 2

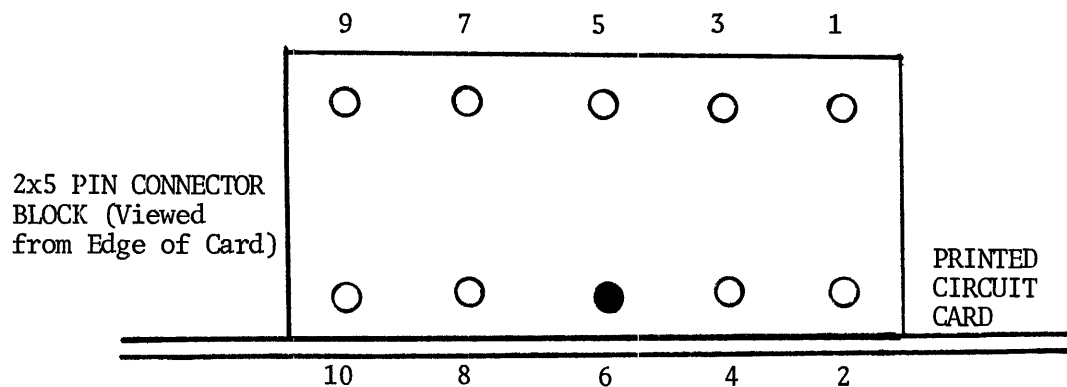
The TU58 is a serial asynchronous peripheral. It is designed to be easily interfaced not only to LSI-11 based systems but also to other computers as well. It may be configured to operate in RS-232C, RS-422 or RS-423 protocol (jumper selectable). Only the DLV11-J currently supports RS-422 on the LSI-11 bus.

<u>FROM</u>	<u>TO</u>	<u>CABLES</u>
{ DLV11 DLV11-E DLV11-F DL11	TU58	Use BC05C-XX (H856 to DB25P) and a BC20N-05 (DB25S to AMP 2x5 pin female)
{ DZV11 DZ11	TU58	Use BC20N-05 (AMP 2x5 pin female to DB25S). RS-232C connection (cable or distribution panel) is supplied with the modules
{ MXV11 DLV11-J	TU58	Use BC20N-05 (AMP 2x5 pin female to DB25S) and a BC21B-05 (DB25P to AMP 2x5 pin female) OR Use BC20M-50 (AMP 2x5 pin female to AMP 2x5 pin female). This is available in 50' lengths only


The pin-out of the serial I/O connector on the TU58 is as follows\*:

<u>PIN #</u>	<u>SIGNAL</u>
1	Auxiliary A
2	Signal Ground
3	TRANSMIT DATA+
4	TRANSMIT DATA-
5	Signal Ground
6	Index in Key - No Pin
7	RECEIVE DATA-
8	RECEIVE DATA+
9	Signal Ground

**digital**  
**COMPONENTS**  
**GROUP**



\* The DECTape II User's Guide incorrectly labels these pins.

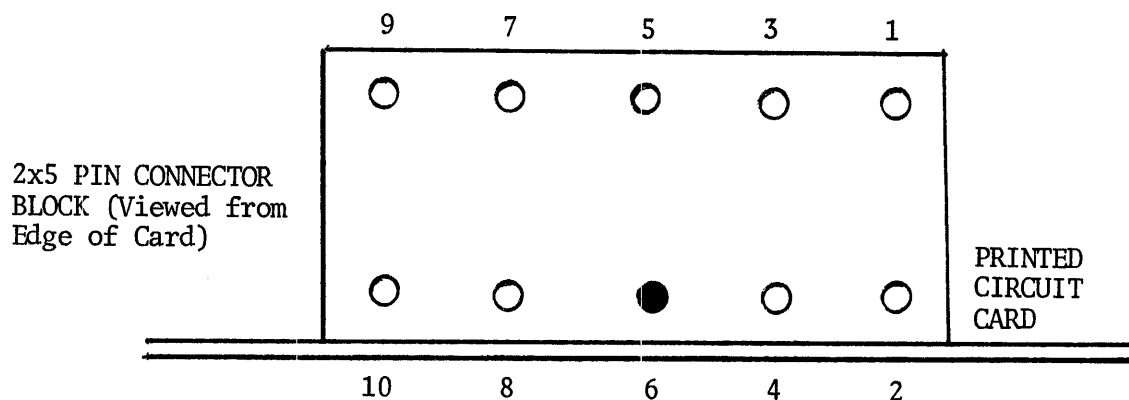
	NUMBER 074
	DATE 7 / 05 / 79
	PRODUCT MXV11-AA, -AC
	PAGE 1 OF 2
TITLE MXV11-AA, -AC Cabling	
DISTRIBUTION Unrestricted	
ORIGINATOR Barbara Beck	

The following DEC cables are recommended for interfacing the MXV11 to a serial device:

- |          |   |
|----------|---|
| BC20N-05 | 5' EIA RS-232C null modem cable to directly interface with an EIA RS-232C terminal (2x5 pin AMP female to RS-232C female)   |
| BC21B-05 | 5' EIA RS-232C modem cable to interface with modems and acoustic couplers (2x5 pin AMP female to RS-232C male)  |
| BC20M-50 | 50' EIA RS-423 or RS-232C cable (2x5 pin AMP female to 2x5 pin AMP female) to connect to another MXV11, DLV11-J, TU58 or any other peripheral with a compatible connector |


The pin-out for the two serial I/O connectors is:

<u>PIN #</u>	<u>SIGNAL</u>
1	UART Clock In or Out (16 x baud rate; CMOS)
2	Signal Ground
3	TRANSMIT DATA+
4	Signal Ground
5	Signal Ground
6	Index in Key - No Pin
7	RECEIVE DATA-
8	RECEIVE DATA+
9	Signal Ground
10	When F1 is installed for the DLV11-KA, +12V is supplied through a 1A fuse to this pin



The DLV11-KA may be used with the MXV11 for EIA to 20mA current loop conversion.

For more information on constructing cables, refer to Micro Note 056.

 TITLE <u>MXV11-A2 Bootstrap Error Halts</u> DISTRIBUTION <u>Unrestricted</u> ORIGINATOR <u>Rich Billig</u>	NUMBER 075
	DATE 7 / 18 / 79
	PRODUCT MXV11-A2
	PAGE 1 OF 3

The MXV11-A2 bootstrap reports errors in operation by HALT instructions. The attached list of HALT addresses (the number displayed by console ODT) details the error indicated by each HALT and the effect of P (PROCEED) in response to the HALT.

TU58 (TAPE) BOOT ERRORS

HALT @ PC

INDICATED ERROR

173076

Memory Diagnostic Failure  
 R2 = Address of Failing Word;  
 Expected Data is Contents of R2  
  
 P = Continue Test

173122

Memory Diagnostic Failure  
 R2 = Address of Failing Word;  
 R3 = Expected Memory Data  
  
 P = Continue Test

173140

Memory Diagnostic Failure  
 — Bad Background Data  
 R2-2 = Address of Failing Word  
 R3 = Expected Memory Data  
  
 P = Continue Test

- 173274 TU58 Controller Error  
R0 = Success Code in Low Byte  
XXX357 = Data Check Error on Medium  
XXX340 = Seek Error (Bad Tape Format)  
XXX337 = Motor Stopped (Drive Malfunction)  
  
P = Restart the Memory Diagnostic
- 173374 Stand-Alone Program Not Found  
  
The file name indicated for stand-alone program loading (by recording it in block #0) or by calling the stand-alone load entry point was not found in the RT11 file directory of the specified unit.  
  
P = HALT Again (Cannot Continue)
- 173440 Protocol Error Between TU58 Controller & LSI-11  
(Serial Line Unit/Cabling Problem)  
  
P = Restart Memory Diagnostic
- 173474 Start Address Invalid  
  
Location 40 in Stand-Alone Program File Does Not Contain a Valid Program Start Address  
  
P = HALT Again (Cannot Continue)

## DISK BOOT ERRORS

- 173056 Memory Diagnostic Failure  
R2 = Address of Failing Word  
R3 = Expected Data  
  
P = CONTINUE Diagnostic
- 173102 Memory Diagnostic Failure  
R2 = Address of Failing Word  
R3 = Expected Data  
  
P = CONTINUE Diagnostic



173270

RL01 Bootstrap Failure

Unit 0 Contains a Bootable Medium but  
Failed to Boot

P = Restart (Retry) RL01 Bootstrap




<b>μnote</b>		NUMBER 077
TITLE <u>Summary of Bootstrap Sources</u>		DATE 8 / 10 / 79
DISTRIBUTION <u>Unrestricted</u>		PRODUCT Bootstraps
ORIGINATOR <u>Charles Giorgetti</u>		PAGE 1 OF 1

The purpose of this Micro Note is to provide a brief summary of the sources that will provide a bootstrap for a given device. Included is a reference to a particular bootstrap that can be entered under the console monitor ODT for a given device.

1. REV11-A,-C bootstrap code is incompatible with the LSI-11/23.
2. The MXV11-A2 chips can only be used with the MXV11-AA,-AC.
3. Revision A chips must be installed on the BDV11 to boot the RX02.
4. The 'L' command under the console monitor ODT is not available with the LSI-11/23.

DEVICE \ SOURCE	CONSOLE MONITOR ODT	REV11-A, -C	BDV11	MXV11-A2
RX01	See Microcomputer Handbook Series	Yes	Yes	Yes
RX02	See Micro Note #065	No	Yes	Yes
TU58	See Micro Note #062A	No	No	Yes
RL01	See Micro Note #063	No	Yes	Yes
MRV11-C	No	No	No	Yes
RK05	See Microcomputer Handbook Series	Yes	Yes	Yes
Paper Tape	See PDP-11 Programmer's Card "L" Command	Yes	No	No

	NUMBER 078
	DATE 8 / 02 / 79
	PRODUCT KDF11-A
	PAGE 1 OF 2
TITLE LSI-11/23 Processor Differences	
DISTRIBUTION Unrestricted	
ORIGINATOR Barry Maskas	

The following facts concerning the LSI-11/23 processor should be carefully considered by all KDF11 users.

1. Sunset Loops: The LSI-11/23 processor has the potential for several microcode loops which cannot be broken out of by using the HALT switch; i.e., it is an infinite loop. Only by negating the BDCOK H for a minimum of 1 microsecond or cycling power off and on can you break the loop. These loops revolve around the general problem that if anything is wrong with memory page zero, any double abort (bus errors, memory management aborts, parity errors, stack overflow) could cause a sunset loop. Because these service inputs have a higher priority than the HALT switch, the HALT switch is never recognized. The three most likely causes of sunset loops which a user may encounter are:
  - a) If the memory management is enabled and page zero of the kernel space is set up as no access; i.e., Status Register Zero (SR0) bits 3-1 contain zeros and Page Description Register (PDR) bits 2 and 1 are zeros, then any trap, interrupt or an abort will cause a read reference to kernel page zero which will cause a memory management abort. The 11/23 will get caught in the loop of trying to service the memory management abort by reading the vector at 250 (octal) which causes another memory management abort.
  - b) If there is no memory in locations 0-377 (octal) or the memory there is faulty, then on traps, interrupts or memory management aborts a read reference to kernel page zero will occur and will cause repeated bus time-outs.
  - c) If memory in locations 0-377 (octal) is ROM, then any kind of double bus error; i.e., the SP (R6) points to non-existent memory during a bus time-out trap, which forces the kernel stack pointer to 4, will cause an infinite loop because of the bus time-outs on the PC push during the trap sequence. The memory will time-out because it does not respond to writes.

2. Stack: The LSI-11/23 has a hardware stack limit check to make sure that the kernel stack does not go below 400 (octal). Since this check does not exist on the LSI-11, the LSI-11/23 will trap to 4 on a stack push below 400 (octal).
3. Masking Interrupts: The line time clock is hardwired to interrupt level 6 and all current I/O devices reside on priority level 4; hence, to mask out all interrupts, a MTPS #340 should be used instead of a MTPS #200.
4. Instructions: Some fundamental differences exist between the way the LSI-11/23 treats some double operand instructions and the treatment by the LSI-11 and the PDP-11/34 when both operands use the same register and the first addressing mode is register (mode 0) and the second addressing mode is auto-increment or auto-decrement (modes 2,3,4 or 5). Since the destination address is calculated before the register operand fetch on the LSI-11/23 and after the register operand fetch on the LSI-11 and PDP-11/34, the effective address may differ between the two machines. This will result in different values being moved for instructions like:

```
MOV R0, (R0)+  
MOV R0, -(R0)  
MOV R1, @-(R1)  
MOV R1, @(R1)+  
MOV PC, @X(R)
```

See Micro Note #053 for PDP-11 Family differences.


<b>μnote</b>		NUMBER 079
TITLE <u>The LSI-11/23 and the LSI-11/2 Buses Are the Same!</u>		DATE 8 / 13 / 79
DISTRIBUTION <u>11/23 Upgraders</u>		PRODUCT LSI-11/23
ORIGINATOR <u>Joe Austin</u>		PAGE 1 OF 1

The LSI-11/23 uses the same Q-bus as the LSI-11/2 and the older quad LSI-11. Ever since it was first developed, the DIGITAL backplanes for the Q-bus were built with room-to-grow; the LSI-11/23 uses this room in order to implement extended memory and priority interrupts. Bus signals that were listed as "spares reserved for DIGITAL use" in earlier documentation have been renamed to reflect their use with the 11/23, as shown in Table 1.

TABLE 1 -- BACKPLANE PIN ASSIGNMENT COMPARISON

LINE	BACKPLANE NAME	KDF11-AA	KD11-F	KD11-HA
AA1	BSPARE1	BIRQ5L	Reserved*	Reserved*
AB1	BSPARE2	BIRQ6L	Reserved*	Reserved*
BP1	BSPARE6	BIRQ7L	Reserved*	Reserved*
AC1	BAD16	BDAL16L	Reserved*	Reserved*
AD1	BAD17	BDAL17L	Reserved*	Reserved*
AE1	SSPARE1	Single Step	Not Used	STOP L
AF1	SSPARE2	SRUNL	SRUNL	SRUNL
AH1	SSPARE3	SRUNL	Not Used	SRUNL
AK1	MSPAREA	Not Used	Not Used	MTOEL
AL1	MSPAREA	Not Used	Not Used	GND
AM2	BIAKIL	MMU STRH	Not Used	Not Used
AR1	BREFL	Not Used†	BREFL	Not Used†
AR2	BDMGIL	UBMAAPL	Not Used	Not Used
BC1	SSPARE4	MMU DAL18H	Not Used	SCLK3H
BD1	SSPARE5	MMU DAL19H	Not Used	SWMIB18H
BE1	SSPARE6	MMU DAL20H	Not Used	SWMIB19H
BF1	SSPARE7	MMU DAL21H	Not Used	SWMIB20H
BH1	SSPARE8	CLK DISL	Not Used	SWMIB21H
BK1	MSPAREB	Not Used	4K RAM BIAS	Not Used
BL1	MSPAREB	Not Used	4K RAM BIAS	Not Used

\* Even though these lines are not used on the KD11-F and KD11-HA, they are based on the backplane and terminated for future bus expansion.

 TITLE <u>LSI-11/23 I/O PAGE ADDRESSING</u> DISTRIBUTION <u>UNRESTRICTED</u> ORIGINATOR <u>RICK PLUMMER</u>	NUMBER 80A
	DATE 10 / 29 / 79
	PRODUCT KDF11-A
	PAGE 1 OF 2

THIS MICRO NOTE SUPERSEDES #80

THE FACTS

In order to maintain a high degree of system compatibility between LSI-11/23 systems and their LSI-11/2 counterparts, the I/O page addressing scheme has been modified. The modification has to do with the physical address which is present on BDAL lines during an I/O page reference.

On revision A3 and earlier KDF11-A modules, BDAL 16 and 17 were forced-active whenever the processor made reference to the I/O page (i.e. whenever BBS7 was asserted). (The revision of the module is stamped into the plastic module handle.) This made it impossible to reference the area between 56KB and 60KB when using the MSV11-D series of memory modules in an unmapped system.

On revision A4 and above, as well as C0 and above, this has been changed. In unmapped mode, the processor will reference the I/O page by asserting a physical address between 56KB and 64KB on the 18-bit address bus, as well as asserting BBS7. In mapped mode, the processor will behave as it has in the past and the I/O page will have a physical address corresponding to 248KB to 256KB present on the bus during BBS7 references.

When using ODT to reference the I/O page with either module, you must still use 18-bit addresses (i.e. 760000 to 777777).

THE IMPLICATIONS

The above has a number of implications with respect to the use of the LSI-11 bus as follows:

1. Devices which should respond as I/O page, whether they are I/O register devices or others such as bootstrap ROM, should respond based on BBS7 and not use BDAL 13-17.

2. Memory devices designed to respond in the 56KB to 64KB memory area should not reply to addresses in this area if BBS7 is asserted. This is especially critical in mapped systems because they power up in unmapped mode and RAM memory will overlay the I/O register. The disabling on BBS7 may optionally be conditioned with BDAL 12 if one half of the I/O page is to be used for memory. This is the case in 60KB systems using the MSV11-D series memories.
3. Other bus masters, such as DMA devices should assert BBS7 if DMA access to the I/O page is desired.

## SPECIFIC DEVICE CONSIDERATIONS

### MSV11-CD MEMORIES

These memories do not have provisions for deselecting on BBS7 and must not be configured to respond to addresses between 56KB and 64KB, or between 248KB and 256KB. They do, however, decode the 18-bit addresses and may be used elsewhere in the system.

### MRV11-C

#### a. Bootstrap Addressing


The bootstrap mode responds only to BBS7 and low order address bits and may be used as a normal bootstrap in the I/O page.

#### b. Direct addressing

The direct address decodes 18-bits but does not deselect on BBS7, therefore, do not configure it for 56KB to 64KB in systems which will be used in mapped mode. In unmapped systems it may be configured for part of the I/O page.

#### c. Window Mapped Mode

The address of the window is decoded using 18-bits but does not deselect on BBS7, therefore, do not configure it for 56KB to 64KB in systems which will be used in mapped mode. It may, however, and quite often is, configured for the bottom 4KB of the I/O page in unmapped systems.

	NUMBER 081
	DATE 8 / 23 / 79
	PRODUCT RXV21, RX02, RX01
	PAGE 1 OF 1
TITLE <u>Use of Recommended Diskettes</u>	
DISTRIBUTION <u>RX02, RX01 Customers</u>	
ORIGINATOR <u>Mark Snyder</u>	

PROBLEM

Users have experienced block errors on RX02's when using diskettes that have normally been used on RX01's. (This problem is not unique to the RX02 and has also been noted when using RX01's.)


EVALUATION

Media reliability is uncertain when using diskettes other than those recommended by DIGITAL. Customers buying cheap, low quality diskettes will frequently encounter unreliable media.

SOLUTION

DIGITAL recommends the use of only DIGITAL or IBM diskettes with the RX01 or RX02. This is especially true where the diskette is used frequently, such as a system device. The following note is repeated from Section 2.3.5 of the RX02 User's Guide.

NOTE: Removable media involve use, handling, and maintenance which are beyond DIGITAL's direct control. DIGITAL disclaims responsibility for performance of the equipment when operated with media not meeting DIGITAL specifications or with media not maintained in accordance with procedures approved by DIGITAL. DIGITAL shall not be liable for damages to the equipment or to media resulting from such operation.

	NUMBER 082
	DATE 8 / 27 / 79
	PRODUCT Serial Printers
	PAGE 1 OF 11
TITLE <u>Handlers for Serial Line Printers</u>	
DISTRIBUTION <u>Unrestricted</u>	
ORIGINATOR <u>Barry Maskas</u>	

THE FOLLOWING IS A SUMMARY OF METHODS FOR PERMANENTLY MODIFYING THE RT-11 V03.00B LP HANDLER TO SUPPORT SERIAL LINE PRINTERS VIA A DLV11 FAMILY INTERFACE.

METHOD ONE

TO SUPPORT LA34 OR LA36 ASCII TERMINALS:

- 1.DETERMINE THE XBUF CSR ADDRESS AND VECTOR
- 2.PERFORM THE FOLLOWING MONITOR SET COMMANDS:

```
.SET LP CR
.SET LP LC
```

- 3.RUN THE PATCH UTILITY:

```
.R PATCH
FILE NAME---
```

```
*LP.SYS
```

```
*1000/ 200
```

```
XCSR VECTOR <CR>
```

NOTE: Enter new

```
*1054/ 177514
```

```
XCSR ADDRESS <CR>
```

XCSI vector and

```
*1204/ 177516
```

```
XBUF ADDRESS <CR>
```

address and XBUF

```
*E
```

address.

THE MODIFIED HANDLER CAN BE COPIED TO OTHER SYSTEM DISKS WITH THE COMMAND:

```
COPY/PRE/SYS LP.SYS DEV:
```

NOTES: EVEN WITH THE DLV11 CONFIGURED AT THE LINE PRINTER ADDRESS THE ABOVE PATCH IS NECESSARY. WHEN INTERFACED IN THIS MANNER, THE LP HANDLER CANNOT DETECT POWER DOWN, OFF LINE, OR PAPER OUT CONDITION; OUTPUT SENT TO THE PRINTER UNDER THESE CONDITIONS WILL BE LOST WITHOUT WARNING.

THE STANDARD LP HANDLER DOES NOT PROVIDE XON/XOF HANDSHAKING OR FORM FEED EMULATION. THE FORMER MUST BE USED IN CONJUNCTION WITH LA120,LS120 AND SERIAL LA180 IF IT IS TO BE RUN AT ITS MAXIMUM SPEED(>2400 BAUD; 9600 BAUD RECOMMENDED FOR LA120 AND LA180, 4800 BAUD FOR LS120). THE LATER MAY BE USED WITH AN LA34,LA35 OR LA36 IF THE LP HANDLER IS TO EMULATE THE FORM FEED FUNCTION NOT PRESENT ON THESE TERMINALS. BOTH OF THESE OPTIONS CAN BE REALIZED WITH THE FOLLOWING LP HANDLER.

**digital**  
**COMPONENTS**  
**GROUP**



## METHOD TWO

TO ASSEMBLE AND INSTALL THE LINE PRINTER DRIVER ON YOUR SYSTEM THE FOLLOWING IS NECESSARY:

- 1.REMOVE THE OLD LP HANDLER I.E., 'REM LP:'
- 2.DELETE THE OLD LP HANDLER I.E., 'DEL/NOQ/SYS LP.SYS'
- 3.EDIT THE FILE: SYCND.MAC, TO REFLECT YOUR SINGLE PRINTER SUPPORT INFORMATION I.E., EITHER LA180 OR LA35, AND THEIR ADDRESS/VECTOR.
- 4.RE-ASSEMBLE AND LINK THE DRIVER. REFER TO THE ENCLOSED INDIRECT COMMAND FILE: LPDRV.COM
- 5.INSTALL LP.SYS VIA: 'INS LP:'

```
;          LPDRV.COM  
;          *****
```

```
MAC/LIST:SY:LP/OBJ:LP SYCND+LP  
LINK/EXE:LP.SYS LP  
DEL/NOQ LP.OBJ
```

## SYCND.MAC

```
.SBTTL SYSTEM CONDITIONAL FILE  
SYSG$N = 1  
TIME$R = 1  
RDF$L = 1  
CLOCK = 60.  
STAR$T = 1  
PWF$L = 1  
ESC$P = 0  
$DIYSYS = 1  
DY$CSR = 177170  
DY$VEC = 264  
L$$180 = 1  
LP$CSR = 177514  
LP.$ = 177514  
LP$VEC = 204  
LP.VEC = 204
```

This SYCND.MAC file is provided as an example. Your system device will contain a file with this title if you SYSGEN'd, and you should reference it in the above procedures.



THE ABOVE SYNCND FILE IS FOR LA180 OR LS120 SUPPORT  
IT CAN BE EDITED TO REFLECT LA3X SUPPORT USING THE  
FOLLOWING SYMBOL DEFINITIONS:

- L\$\$180 - DEFINE THIS SYMBOL IF YOU WISH LA180 OR LS120 SUPPORT
- L\$\$35 - DEFINE THIS SYMBOL IF YOU WISH LA35, LA36 OR LA30 SUPPORT
- M\$\$RGN - USED IN CONJUNCTION ONLY WITH L\$\$35 SYMBOL TO DEFINE THE MARGIN TO BE USED BETWEEN PAGES. IF NOT SPECIFIED, DEFAULT TO SIX
- LP\$CSR - XBUF CSR ADDRESS, THE DEFAULT IS STANDARD LP CSR ADDRESS=177514
- LP\$VEC - XBUF VECTOR ADDRESS, THE DEFAULT IS STANDARD LP VECTOR=200
- LP.CSZ - THE COLUMN WIDTH OF THE OUTPUT DEVICE IF NOT SPECIFIED, DEFAULT IS 132. COLUMNS. FOR LA30 SUPPORT SET THIS SYMBOL TO 80. OR USE THE 'SET LP: WIDTH=80' COMMAND

THE FOLLOWING KMON 'SET' COMMAND IS AVAILABLE WHEN THE LINE  
PRINTER IS ASSEMBLED FOR LA35 SUPPORT:

SET LP: PAGE=NN ;SET PAGE SIZE NOT INCLUDING MARGIN  
THE DRIVER DEFAULT VALUE IS 61. THE TARGET IS THAT THE VALUE  
OF M\$\$RGN (DEFAULT 6) PLUS THE PAGE SIZE (DEFAULT 61) MINUS ONE IS  
THE SIZE OF A PAGE IN LINES (DEFAULT RESULTING VALUE IS 66.).

```

      .TITLE LP      V03.03
      .IDENT /V03.03/
;FZ033---ADDED LA180S (AND LS120) SUPPORT
;
;      ADDED LA35 (AND LA30,LA36) SUPPORT
;ADDITIONAL AMENDMENTS FOR XON AND XOF INCLUDED
;RT-11 LINE PRINTER (LP/LS11) HANDLER
;
;
;COPYRIGHT (C) 1978
;
;DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS 01754
;
;THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY
;ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH
;THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE,
;OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE
;AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO
;ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE
;SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.
;
;THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE
;WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A

```



```
;COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.  
;  
;DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR  
;RELIABILITY OF ITS SOFTWARE ON EQUIPMENT  
;WHICH IS NOT SUPPLIED BY DEC.
```

```
.MCALL .DRBEG,.FORK,.DREND,.DRAST,.DRFIN,.QELDF
```

```
;SYSTEM GENERATION OPTIONS:
```

```
.IIF NDF MMG$T, MMG$T=0  
.IIF NDF ERL$G,ERL$G=0  
.IIF NDF TIM$IT, TIM$IT=0
```

```
.QELDF
```

```
;DEFAULT CONTROL REGISTER DEFINITIONS
```

```
.IIF NDF LP$CSR, LP$CSR==177514 ;STANDARD LP CONTROL REGISTER  
.IIF NDF LP$VEC, LP$VEC==200 ;STANDARD LP VECTOR
```

```
LPDSIZ =0 ;DEVICE BLOCK SIZE (0 IF NONFILE-STRUCTURED)  
.IIF NDF LPSTS,LPSTS=20003 ;DEVICE STATUS (WRITE ONLY)
```

```
;CONSTANTS FOR MONITOR COMMUNICATION
```

```
HDERR =1 ;HARD ERROR BIT  
V.MAX =500 ;MAXIMUM VECTOR
```

```
;ASCII CONSTANTS
```

```
CR =15  
LF =12  
FF =14  
HT =11
```

```
.IF DF L$$180
```

```
CNTRLQ =21 ;XON CHARACTER  
CNTRLS =23 ;XOFF CHARACTER  
.ENDC
```

```
;DEFAULT COLUMN SIZE DEFINITION
```

```
.IIF NDF LP.CSZ,LP.CSZ=132. ;DEFAULT IS 132 COLUMNS  
COLSIZ ==LP.CSZ  
.IF DF L$$35  
;DEFAULT MARGIN SIZE IF NOT SPECIFIED  
.IIF NDF M$$RGN, M$$RGN = 6.  
.ENDC
```



‡THE FOLLOWING ARE THE PARAMETERS FOR INTERFACE TO THE MONITOR "SET"  
‡COMMAND

.ASECT

.=400

.IF DF L\$\$35

.WORD M\$\$RGN ‡MINIMUM PAGE HEIGHT

.RAD50 /PAGE /

.WORD <0.PAGE-400>/2+40000 ‡NO 'NO' OPTION, NUMBER REQUIRED

.ENDC

.WORD 30. ‡MINIMUM WIDTH

.RAD50 /WIDTH /

.WORD <0.WIDTH-400>/2+40000 ‡NO 'NO' OPTION, NUMBER REQUIRED

NOP ‡NO CR => NOP CROPT

.RAD50 /CR /

.WORD <0.CR-400>/2+100000 ‡ALLOW 'NO'

NOP ‡NO FORM0 =>NOP FFOPT

.RAD50 /FORM0 /

.WORD <0.FORM0-400>/2+100000 ‡ALLOW 'NO'

BMI LPERR-ERROPT+. ‡NO HANG BY GOING TO ERROR

.RAD50 /HANG /

.WORD <0.HANG-400>/2+100000 ‡ALLOW 'NO'

.WORD 40 ‡FOR NO LC, CONVERT LC TO UC

.RAD50 /LC /

.WORD <0.LC-400>/2+100000

BNE IGNORE-CTROPT+. ‡IGNORE CTRL CHAR IF NOCTRL

.RAD50 /CTRL /

.WORD <0.CTRL-400>/2+100000

BEQ TABSET-TABOPT+. ‡SIMULATE TAB IF NOTAB

.RAD50 /TAB /

.WORD <0.TAB-400>/2+100000

.WORD 0 ‡END OF LIST

‡+

‡ \*\*=0.?????

‡

THE FOLLOWING 'SUBROUTINES' SERVICE THE 'SET' COMMANDS  
TYPED TO KMON TO ALTER SEVERAL OPTIONS IN THE DRIVER (  
SUCH AS PAGE HEIGHT, COLUMN WIDTH, 'WRITE-ALL' MODE, ETC.)  
;-

```

O.WIDTH:MOV      R0, COLCNT      ;NEW WIDTH TO 2 CONSTANTS
MOV      R0, RSTC+2
        .IF DF L$$35
MOV      R0, VTLOOP+2        ;NEED IT IN 3 PLACES HERE
        .ENDC
CMP      R0, R3              ;ERROR IF <30.
RTS      PC

        .IF DF L$$35
O.PAGE:  MOV      R0, $PAGE      ;SET PAGE SIZE
MOV      R0, $HOLD            ;INTO TWO PLACES
CMP      R0, R3              ;BUT DON'T ALLOW NUMBERS LESS M$$RGN
RETURN
        .ENDC

O.CR:    MOV      (PC)+, R3      ;NO 'NO', SO SET TO DO CR
BEQ      RSTC-CROPT+,
MOV      R3, CROPT           ;SET CR OPTION
RTS      PC

O.FORMO:MOV      (PC)+, R3      ;SET TO DO FORMFEEDS ON BLOCK 0
BEQ      BLKO-FFOPT+,
MOV      R3, FFOPT
RTS      PC

O.HANG:  MOV      (PC)+, R3      ;SET TO HANG
BMI      RET-ERROPT+,
MOV      R3, ERROPT
RTS      PC

O.LC:    CLR      R3            ;FOR 'LC', LEAVE LOWER CASE STUFF ALONE
NOP
MOV      R3, LCOPT
RTS      PC

O.CTRL:  MOV      (PC)+, R3      ;SET TO PASS ALL NON-PRINTING CHAR
BNE      PC1-CTROPT+,
MOV      R3, CTROPT
RTS      PC

O.TAB:   MOV      (PC)+, R3      ;PASS TAB DIRECTLY TO LF
BEQ      HDWTAB-TABOPT+,
MOV      R3, TABOPT
RTS      PC
    
```

```

;LOAD POINT
;IF NDF L$$180 ;NO TABLE IF NOT DEFINED
.DRBEG LP,LP$VEC,LPDSIZ,LPSTS
;IFF ;OTHERWISE INCLUDE TABLE FOR 2 VECTORS
.DRBEG LP,LP$VEC,LPDSIZ,LPSTS,LPTAB
.ENDC

;ENTRY POINT

MOV LPCQE,R4 ;R4 POINTS TO CURRENT Q ENTRY
ASL 6(R4) ;WORD COUNT TO BYTE COUNT
;IF NDF L$$180
BCC LPERR ;A READ REQUEST IS ILLEGAL
BEQ LPDONE ;SEEKS COMPLETE IMMEDIATLY
;IFF
BCS 98$
JMP LPERR
98$: BNE 99$
JMP LPDONE
99$:
.ENDC
RET:
;IF DF L$$180
MTPS #340
TST XOF
BNE 1$
;IFTF
BIS #100,@LPS ;CAUSE AN INTERRUPT,STARTING TRANSFER
;IFT
1$: MTPS #0
BIS #100,@#LP$CSR-4 ;INTERRUPT ENABLE INPUT SIDE

.ENDC
RTS PC

;IF DF L$$180
;
;XON/XOFF REQUIRE TWO VECTORS AND ISR'S, INCLUDE THE TABLE
;
LPTAB:
.WORD LP$VEC-4 ;OTHER IS INPUT SIDE
.WORD LRINT-, ;OFFSET TO INPUT ISR
.WORD 340 ;PR7
.WORD LP$VEC ;FIRST IS OUTPUT VECTOR
.WORD LPINT-, ;OFFSET TO IT'S ISR
.WORD 340 ;PR7
.WORD 0 ;E-O-T

```

```

;+
;
;**-LRINT
;INPUT SIDE INTERRUPT SERVICE ROUTINE. IF WE RECEIVE AN XON
;WE SET THE INTERRUPT ENABLE ON THE OUTPUT SIDE. IF WE RECEIVE
;AN XOFF, WE TURN OUTPUT I.E. OFF. IF A JUNK CHARACTER IS SEEN
;WE THROW IT AWAY AND ASSUME THE USER IS TRYING TO TYPE WHILE
;WE ARE.
;--
.DRAST LR,4,LRDONE      ;;;FAST ENTRY POINT
MOV    @#LP$CSR-2,R4    ;;;PICK UP CHARACTER TYPED
BIC    #177600,R4      ;;;INSURE WE GET ONLY 7-BITS
CMP    #CNTRLQ,R4      ;;;WAS THIS AN XON?
BNE    2$              ;;;IF EQ YES, ENABLE OUTPUT SIDE
CLR    XOF
BR     RET
2$:    CMP    #CNTRL5,R4 ;;;WAS THIS AN XOFF?
BNE    1$              ;;;IF NE NO, IGNORE CHARACTER AND GO
MOV    #1,XOF
CLR    @#LP$CSR        ;;;IF EQ YES,TURN OUTPUT SIDE OFF
1$:    BIS    #100,@#LP$CSR-4 ;;;ALWAYS RE-ENABLE SELF
RTS    PC              ;;;AND RETURN TO MONITOR
RDONE: CLR    @#LP$CSR-4 ;;;INTERRUPT DISABLE ONLY
RTS    PC              ;;;
.ENDC

;+
;**-LPINT
;
;THIS IS THE OUTPUT INTERRUPT SERVICE ROUTINE. PICK UP CURRENT
;PACKET, GET-BYTE, AND SEE IF CHARACTER REQUIRES SPECIAL PROCESS
;SUCH AS THE FF AND TAB.
;--

.ENABL LSB
.IF EQ MMG$T
.IFTF
.DRAST LP,4,LPDONE
MOV    LPCQE,R4        ;R4 ->CURRENT QUEUE ELEMENT
.IF    NDF             L$$180
TST    @(PC)+          ;ERROR CONDITION?
.IFF
CLN
BR     ERROPT
.ENDC
LPS:  .WORD    LP$CSR    ;LINE PRINTER STATUS REGISTER

```

```

ERROPT: BMI      RET      ;YES-HANG TILL CORRECTED
        TSTB     @LFS     ;IS IT READY
        BFL      RET      ;NO,RETURN
        CLR      @LFS     ;YES, DISABLE INTERRUPTS
        .FORK    LPFBLK   ;REQUEST A SYSTEM PROCESS
        TST      (R4)     ;IS THIS BLOCK 0?
FFOPT:  BEQ      BLK0     ;YES-OUTPUT INITIAL FORM FEED
LPNEXT: TSTB     @LFS     ;READY FOR ANOTHER CHAR YET?
        BFL      RET      ;NOPE-RETURN FROM INTERRUPT
        ASLB     (PC)+    ;TAB IN PRIGRESS?
TABFLG: .WORD    0
        BNE     TAB      ;BRANCH IF DOING TAB

        .IF DF L$$35
        DEC     $SKIP    ;AN I IN MIDDLE OF LF STREAM?
        BGT     VTLOOP   ;IF GT YES, CONTINUE TILL DONE
        .ENDC

IGNORE: TST      6(R4)    ;ANOTHER CHARACTER TO TRANSFER?
        BEQ     LPDONE   ;BR IF NOT, XFER DONE
        .IFT
        MOVB    @4(R4),R5 ;GET NEXT CHAR (IF ANY)
        INC     4(R4)    ;BUMP BUFFER POINTER

        .IFF
        JSR     PC,@$GTBYT ;GET A BYTE FROM USER BUFFER
        MOV     (SP)+,R5  ;PUT IN R5
        .IFTF
        INC     6(R4)    ;BUMP CHARACTER COUNT
        BIC     #177600,R5 ;7-BIT
        CMPB    #40,R5   ;PRINTING CHAR?
        BHI     CHRTST   ;NO-GO TEST FOR SPECIAL CHAR.
        CMPB    #140,R5  ;LOWER CASE?
        BHIS    PCHAR    ;NO
        SUB     (PC)+,R5 ;YES, CONVERT IF DESIRED

LCOPT:  40
PCHAR:  DEC     (PC)+    ;ANY ROOM LEFT ON LINE?
COLCNT: .WORD   COLSIZ  ;# OF PRINTER COLUMNS LEFT
        BLT     IGNORE   ;NO MORE ROOM ON LINE, DON'T PRINT CHAR
        ASLB    (PC)+    ;UPDATE TAB COUNT
TABCNT: .WORD   1
        BEQ     RSTTAB   ;RESET TAB
PC1:    MOVB    R5,@(PC)+ ;PRINT THE CHAR
LPB:    .WORD   LP$CSR+2 ;LINE PRINTER BUFFER REGISTER
        BR      LPNEXT   ;TRY FOR NEXT CHAR
CHRTST: CMPB    #HT,R5   ;IS CHAR A TAB?

```



```

TABOFT: BEQ     TABSET           ;YES-RESET TAB
        CMPB    #LF,R5          ;IS IT LF?
        .IF DF L$$35
        BEQ     PGCNT           ;IF EQ YES,COUNT NUMBER OF LINES
        .IFF
        BEQ     RSTC            ;YES-RESTORE COLUMN COUNT
        .ENDC
        CMPB    #CR,R5          ;IS IT CR?
        .IF DF L$$180 ! L$$35
CROFT:  BEQ     RSTC            ;REASONABLE DEFAULT FOR LA35/LA180S
        .IFF
CROFT:  NOP                     ;IGNORE UNLESS MODIFUED
        .ENDC
        CMPB    R5,#FF          ;IS IT A FF?

        .IF DF L$$35
        BEQ     FFFLT           ;IF EQ YES,SPACE UP PAPER RIGHT
        .ENDC

CTROFT: BNE     IGNORE          ;NO-CHAR IS NON-PRINTING
        .IF DF L$$35
PGCNT:  DEC     $PAGE           ;KEEP TRACK OF NUMBER OF LINES
        BEQ     PGFLT           ;IF EQ THEN PAGE FAULT, DO SKIPS
        .ENDC

RSTC:   MOV     #COLSIZ,COLCNT  ;RE-INITIALIZE COLUMN COUNTER
RSTTAB: MOV     #1,TABCNT       ;RESET TAB COUNTER
        BR      PC1             ;PRINT THE CHAR

TABSET: MOV     TABCNT,TABFLG   ;SET UP TAB
TAB:    MOV     #40,R5          ;PRINT SPACES
        BR      PCHAR

HDWTAB: ASLB    TABCNT          ;ADJUST TAB COUNT
        BEQ     RSTTAB          ;IF EQ, RESET TAB COUNT
        DEC     COLCNT          ;ELSE ADJUST COLUMN COUNT
        BR      HDWTAB         ;CONTINUE UNTIL NEXT TAB POSITIO

        .IF DF L$$35
FFFLT:  MOV     $PAGE,$SKIP     ;NUMBER OF LINES LEFT IN PAGE
        ADD     #M$$RGN-1,$SKIP ;PLUS SKIPPING OVER THE BOUNDARY
        MOV     $HOLD,$PAGE     ;RESET NUMBER LINES/PAGE
        BR      VTLOOP          ;GENERATE CORRECT NUMBER OF LF'S
PGFLT:  MOV     $HOLD,$PAGE     ;RESET LINES/PAGE COUNTER
        MOV     #M$$RGN,$SKIP   ;DO M$$RGN LINES BETWEEN PAGES
VTLOOP: MOV     #COLSIZ,COLCNT  ;RESET FORM WIDTH
        MOV     #1,TABCNT       ;RESET TAB COUNTER
        MOV     #LF,R5          ;PRINT LINE FEEDS

```

```

BR      PC1          ;UNTIL NO MORE SKIPS TO DO
.ENDC

BLKO:   INC      @R4          ;MAKE SURE WE ONLY COME HERE ONCE
MOV     #FF,R5          ;PRINT INITIAL FF
        .IF DF L$$35
BR      FFFLT          ;SKIP REQUIRED NUMBER OF LF'S
        .IFF
BR      RSTC
        .ENDC
LPERR:  BIS      #HDERR,@-(R4) ;SET HARD ERROR BIT
        .ENDC
        .DSABL   LSB

        ;OPERATION COMPLETE


LPDONE:
        .IF DF L$$180
CLR     @#LP#CSR-4      ;TURN INPUT INTERRUPTS OFF ALSO
        .ENDC
CLR     @LPS           ;TURN OFF INTERRUPT
        .DRFIN   LF

        .IF DF L$$35
$PAGE:  .WORD    61.          ;LINES/PAGE DEFAULT TO THIS
$HOLD:  .WORD    61.          ;LINES/PAGE HOLDER
$SKIP:  .WORD    0.          ;#OF LINES BETWEEN PAGES COUNTER
        .ENDC

LPFBLK: .WORD    0,0,0,0      ;FORK BLOCK
        .IF     DF      L$$180
XOF:    .WORD    0
        .ENDC
        .DREND   LF

        .END

```

 TITLE <u>Alternate Clock Frequencies for the MXV11</u> DISTRIBUTION <u>MXV11 Users</u> ORIGINATOR <u>Charlie Giorgetti</u>	NUMBER 083
	DATE 9 / 05 /79
	PRODUCT MXV11-AA, -AC
	PAGE 1 OF 2

The purpose of this Micro Note is to define frequencies, other than the standard 60Hz., that can be obtained from the MXV11 to be used as the BEVNT clock. The frequencies defined are 50Hz., 300Hz. and 2.4KHz.

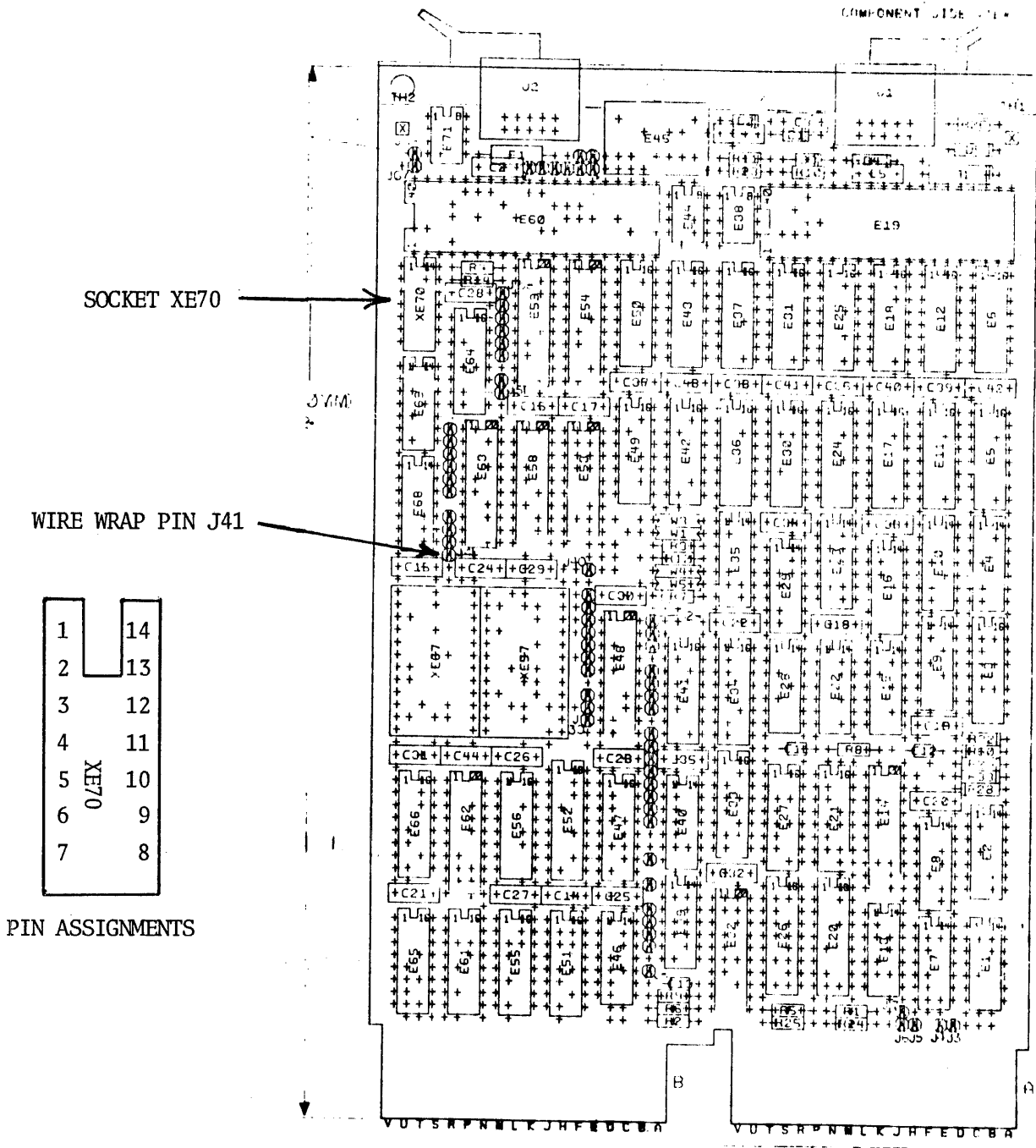
The 50Hz. clock is obtained from the MXV11 by replacing one chip. The chip that must be changed is in an IC socket, making replacement simple. In location XE70 (see attached figure) of the board layout is a 74LS90; it is replaced pin for pin with a 74LS92 to obtain the 50Hz. clock.

To obtain the 300Hz. clock from the MXV11 involves the removal of one chip and connecting two pins left by the removed chip. The chip that is removed is the 74LS90 at location XE70. The removed chip is replaced with a 14-pin component carrier. The component carrier allows pin 1 and pin 8 of the IC socket (see attached figure) to be connected, without damage or permanent alternations being done to the actual board itself.


The 2.4KHz. signal can be obtained from the MXV11 by removing one chip and connecting a wire wrap jumper to a pin location left by the removed chip. The 74LS90 is removed from the IC socket at XE70 and is replaced with the component carrier. Pin location 8 of the carrier is connected to wire wrap pin J41.

#### NOTES

1. The operating systems RT-11 and RSX-11 run at either 50Hz. or 60Hz.
2. Since these are non-standard configuration changes, it is recommended that the MXV11 be returned to its original configuration prior to returning it for repair in order to prevent it from being rejected by the DIGITAL repair center.

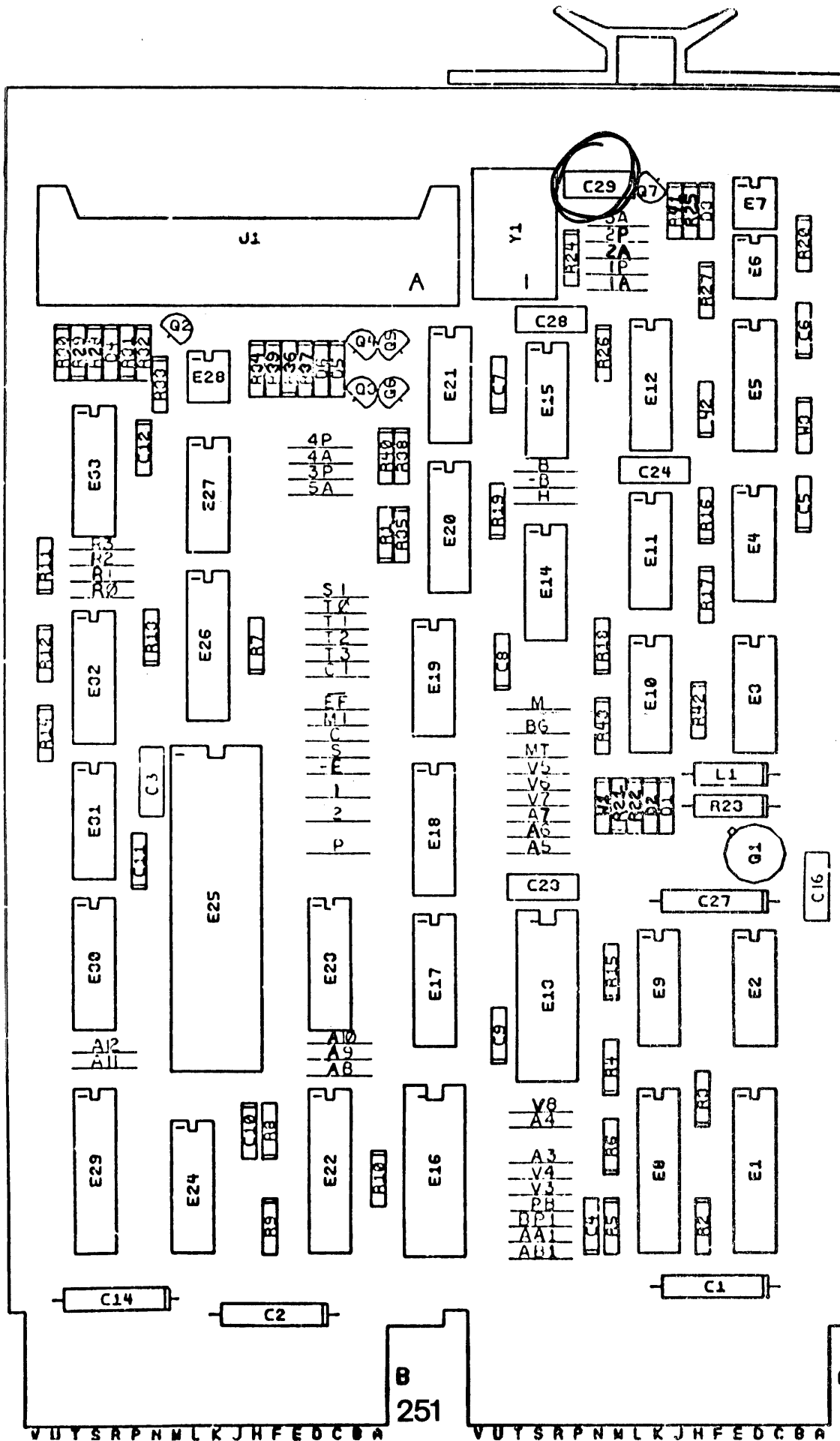


LOCATION AND PIN ASSIGNMENTS OF XE70 ON THE MXV11

		NUMBER 084
		DATE 10 / 11 / 79
TITLE	Improved DLV11-F	PRODUCT DLV11-F
DISTRIBUTION	Unrestricted	PAGE 1 OF 2
ORIGINATOR	Barry Maskas	

The DLV11-F (M8028) has been re-etched due to some improvements in the optical isolation 20ma receive circuit and the -12V charge pump circuit. The logical jumper functions are unchanged and are explained in the 1978-79 Memories and Peripherals Handbook, however, the physical location of these jumpers on a board is shown in the diagram.

To use this board for a 20ma interface with a TTY, ensure that capacitor C29 (0.005UF) is installed as shown in the diagram.



V U Y S R P N M L K J H F E D C B A

V U Y S R P N M L K J H F E D C B A

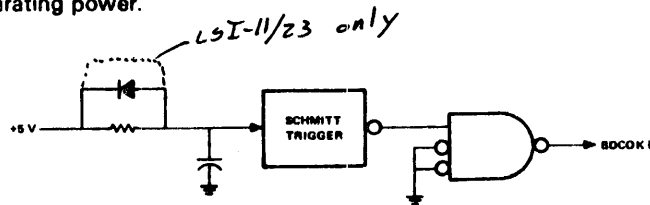
<b>μnote</b>		NUMBER 085
TITLE	WAKE-UP CIRCUIT IMPLEMENTATIONS	DATE 10 / 29 / 79
DISTRIBUTION	UNRESTRICTED	PRODUCT LSI-11/2, LSI-11/23
ORIGINATOR	RICK PLUMMER	PAGE 1 OF 2

### PURPOSE OF THE WAKE-UP CIRCUIT

The wake-up circuit is included on the LSI-11/2 and LSI-11/23 processors in order to allow users to implement small systems without the need for more complex power sensing circuitry. Small systems are those which do not contain disk mass storage devices. Systems containing disks as well as those containing non-volatile read/write memory (i.e. core or battery backed-up RAM) or other devices depending on both DCOK and POK or power failure detection for proper system operation should utilize the full power sequencing implemented with a Digital Power supply, KPV11 module or other means. When using the full power sequencing the processor, on-board wake-up circuit should be disabled. This will be described later in this note.

For reference purposes, the wake-up circuit description from the Microcomputer Processor Handbook is reprinted below.

**Wake-Up Circuit** - The wake-up circuit causes the LSI-11 processor to self-initialize during power-up. An R-C circuit receives +5 V operating power when power is turned on. When power is first applied, the low capacitor voltage causes the Schmitt trigger's output to go high and the bus driver asserts the BDCOK H signal (low). After power has been applied for approximately 1 second, the capacitor's voltage rises above the Schmitt trigger's threshold voltage, and its output goes low. The low voltage turns off the bus driver, enabling BDCOK H to become asserted. The processor then starts its initialization sequence if no other device is asserting BDCOK H. Proper initialization requires that +12 V operating power be applied within 50 ms of +5 V operating power.



Normal operation of the wake-up circuit depends on the rise time of the +5 V power supply being faster than 50 ms. The +12 V power supply also must attain its specified operating voltage in the same 50 ms. The wake-up circuit does not provide power failure detection nor power-down sequencing. These functions, if required, must be generated externally.

**digital**  
**COMPONENTS**  
**GROUP**

The circuit is the same for both processors, although physical implementation is different as described below.

#### LSI-11/2

The wake-up circuit, as shown in the figure, is fully implemented and the module is shipped with the wake-up option fully functional.

To disable the wake-up function, remove capacitor C81 by cutting its leads at the board.

#### LSI-11/23

The wake-up circuit on the LSI-11/23 is the same as shown in the figure. It is shipped with a red wire across diode D1, thus disabling the circuit.

To enable the wake-up circuit, remove the red wire across diode D1 by cutting at the board. Be careful not to damage the diode.

Some early revision modules were shipped without the wire. A later revision will change this wire to a jumper strap at some time in the future.



<b>ynote</b>		<b>NUMBER</b> 086
<b>TITLE</b> <u>INTERFACING TO THE TU58 WITHOUT BREAK</u>		<b>DATE</b> 1 / 09 / 80
<b>DISTRIBUTION</b> <u>UNRESTRICTED</u>		<b>PRODUCT</b> TU58
<b>ORIGINATOR</b> <u>JON TAYLOR</u>		<b>PAGE 1 OF 2</b>

The controller of a TU58 intelligent tape cartridge system can be connected to any serial interface that conforms to RS-422, RS-423 or RS-232C interface standards. This allows a TU58 drive to be connected to any non-DEC host computer that provides these interfaces. The TU58 can be connected to such a host even though its interface is incapable of transmitting a break ("space" condition) to the controller. This article explains the hazards involved with using such an interface.

The TU58 DECtape II User's Guide (EK-0TU58-UG) describes the radial serial protocol that is used by a processor to communicate with a TU58 controller. The protocol uses a break signal to initialize the controller in much the same way as the LSI bus BINIT signal is used to initialize devices on the bus. Regardless of the state of the line protocol (and the controller), the controller will always detect the break signal as it is routed to the controller's "non-maskable interrupt".

Break is normally used in two situations:

1. On power-up, the TU58 continuously sends INIT bytes to the host. The host sends break and two INIT bytes. The TU58 responds with a CONTINUE byte and is ready for use.
2. If communications break down due to a protocol, line, or TU58 error, the host can restore order by sending a break and two INIT bytes. As above, the TU58 will respond with a CONTINUE and wait for further instructions.

In situation one, a host without break capabilities would send just one INIT byte and the TU58 will respond in the usual way with a CONTINUE. The host should be prepared to ignore one or two INIT bytes that may be seen before the CONTINUE byte (due to UART buffering).

The absence of break in the second situation is the cause of less reliable operation.

In most cases, the standard checksumming of messages and protocol handshaking will detect a protocol or line error and the state of the protocol will be known.

**digital**  
**MICROCOMPUTER**  
**PRODUCTS**  
**GROUP**

To reset the TU58, the breakless host would send one INIT byte and wait for a CONTINUE byte response. However, in the case that an error occurs while the host is sending a packet to the TU58, more than one INIT must be sent, as the TU58 can not distinguish the INIT from the packet it is expecting. There is a very slim chance in a write operation (one in 65536) that when the TU58 interprets two of the INIT's as a checksum word, the checksum will actually be correct and an erroneous write will occur.

Very infrequently, communications may break down due to a TU58 controller malfunction (caused by power glitches or noisy environments). Without break, the only way to reset the controller is to power it off and on.

To avoid a possible malfunction because the controller can not be initialized, Digital recommends that all serial interfaces to the TU58 be capable of generating a space condition.

# note

TITLE TYPE-IN BOOTSTRAP FOR TU58  
DISTRIBUTION UNRESTRICTED  
ORIGINATOR CHARLIE GIORGETTI


NUMBER 0087
DATE 12 / 18 / 79
PRODUCT TU58
PAGE 1 OF 1 <i>REP</i>

The following is a short type-in bootstrap for the TU58. This code, using the TU58 bootstrap command reads block 0 from the TU58 drive 0. It then executes from location 0 in memory. This code will boot an RT-11 TU58 system, an RSX-11S System, or suitable user-written software.

To implement the boot, all interrupts must be inhibited. This can be done under ODT, enter \$\$/\_----- 340 (CR). The stack pointer and program counter must be initialized. Under ODT enter R6/\_----- 1000 (LF), R7/\_----- 1000 (CR). Type in the bootstrap starting at 1000 and type P.

001000	012701	176500	START:	MOV #176500,R1	;INIT RCSR ADDR
001004	012702	176504		MOV #176504,R2	;INIT XCSR ADDR
001010	010100			MOV R1,R0	;INIT BREAK COUNTER
001012	005212			INC (R2)	;SET BREAK
001014	105712		WAIT1:	TSTB (R2)	;TEST FOR READY
001016	100376			BPL WAIT1	;WAIT FOR READY
001020	006300			ASL R0	;SHIFT COUNTER
001022	001005			BNE XBYT	;NOT ZERO, XMIT
001024	005012			CLR (R2)	;ZERO, CLEAR BREAK
001026	012700	000004		MOV #4,R0	;SET-UP INIT,BOOT COUNTER
001032	005761	000002		TST 2(R1)	;DUMP RECIEVER BUFFER
001036	042700	000020	XBYT:	BIC #20,R0	;CLEAR AFTER BOOT CODE
001042	010062	000002		MOV R0,2(R2)	;XMIT CHAR
001046	001362			BNE WAIT1	;IF ZERO NOT XMITTED
001050	005003			CLR R3	;ZERO, INIT MEM POINTER
001052	105711		WAIT2:	TSTB (R1)	;TEST FOR READY
001054	100376			BPL WAIT2	;WAIT FOR READY
001056	116123	000002		MOVB 2(R1),(R3)+	;PUSH BYTE INTO MEM
001062	022703	001000		CMP #1000,R3	;512 BYTES RECEIVED
001066	101371			BHI WAIT2	;IF NOT LOOP
001070	005007			CLR PC	;IF YES EXECUTE AT 0

**digital**  
**MICROCOMPUTER**  
**PRODUCTS**  
**GROUP**

	NUMBER
	088
	DATE
	1 / 04 / 80
TITLE <u>RT-11 V3B FB MONITOR AND TU58's</u>	PRODUCT
DISTRIBUTION <u>UNRESTRICTED</u>	TU58
ORIGINATOR <u>JON TAYLOR</u>	PAGE 1 OF 1
	<i>282</i>

Users of RT-11 V3B and TU58's should be aware that I/O errors on TU58 access or total system failure may occur under the following conditions.

1. 11/03 processor
2. Serial line interface to TU58 controller runs at 38.4K baud
3. RT-11 V3B foreground/background monitor

This is due to a timing problem internal to the RT-11 FB monitor.

The May 1979 RT-11 Software Dispatch (AD-C740B-13) contains two binary patches (sequence numbers 20M or 22M) which, when applied, will remedy the problem. In the same issue, note that optional performance improvement patches 210 and 230 apply only to PDT, 11/03, 11/23 and 11/34 processors. Once patched, the software can not be run on other than the above processors.

<b>μnote</b>		<b>NUMBER</b> 089
<b>TITLE</b> <u>STANDALONE PROGRAM LOADER</u>		<b>DATE</b> 3 / 04 / 80
<b>DISTRIBUTION</b> <u>UNRESTRICTED</u>		<b>PRODUCT</b> DEVELOPMENT SYSTEMS
<b>ORIGINATOR</b> <u>JON TAYLOR</u>		<b>PAGE 1 OF 5</b> <i>RBP</i>

This micro note includes a listing of a program loader ("LOADER") that runs as a background job under RT-11. LOADER is useful to customers who are developing and debugging standalone FORTRAN IV and/or MACRO-11 programs (that is, programs which run without operating system support). LOADER will read any memory image file (.SAV file) from any RT-11 supported device into memory (overwriting RT-11), and begin its execution. No hardware bootstraps are required to debug the program.

The .SAV file needs no special processing before loading. The RT-11 assembler and linker cooperate to produce the standalone file with no special commands. For example, consider the following standalone FORTRAN program, called "TEST.FOR":

```

10      IF ((IPEEK("177560").AND."200) .EQ. 0) GOTO 10
        ICHAR = IPEEK("177562) .AND. 255

20      IF ((IPEEK("177564).AND."200) .EQ. 0) GOTO 20
        CALL IPOKE("177566, ICHAR)
        GOTO 10
        END

```

The following commands will produce a standalone image:

```

.FOR TEST           ! compile FORTRAN program
.LINK/EXE:TEST UNI,TEST      ! UNI contains $SIMRT

```

The following command will load TEST.SAV and begin its execution:

```

.R LOADER
*TEST

```

LOADER uses locations 40,42, and 50 in the .SAV file to determine initial PC (start address), initial SP (stack address), and program high limit, respectively.

NOTE: The LOADER algorithm requires that the background partition be large enough to contain both LOADER and the standalone program. Therefore, the total amount of memory required to run LOADER should be at least the sum of the following:

**digital**  
**MICROCOMPUTER**  
**PRODUCTS**  
**GROUP**

- 256 words (vector and stack area)
- size of loader program (about 128 words)
- size of standalone program (from its link map)
- size of any Foreground Job, or System Tasks, USR (if SET NOSWAP), and any loaded handlers
- size of RMON

To maximize the memory available for the standalone program, no Foreground Job or System Tasks should be loaded, the USR should be SET SWAP, and all resident drivers should be UNLOADED.

An undocumented feature of the MXV11-A2 TU58 bootstrap can be used to load standalone programs after they have been debugged using LOADER. As described in the listing contained in Micro Note 62A, the bootstrap can load a program from an RT-11 file-structured TU58 cartridge into memory and begin its execution. The cartridge is prepared in the following manner:

```
.INIT/NOQ DD:           ! initialize the TU58, if required
.COPY TEST.SAV DD:      ! run the standalone program on the TU58
.R PATCH                ! run RT-11 patch program
File Name? DD:          ! and "patch" the TU58
*0/ xxxxxxxx 260 LF     ! put a 260 as first word of first TU58 block
2/ xxxxxxxx 76733 LF   ! RAD50 for "TES"
4/ xxxxxxxx 76400 LF   ! RAD50 for "T"
6/ xxxxxxxx 73376 CR   ! RAD50 for "SAV"
*E CR
```

Now when the TU58 is booted by the MXV11-A2, TEST.SAV will begin execution.

```

1          .TITLE  LOADER
2          .IDENT  /MAY.80/
3
4          .ENABL  LC
5
6          .MCALL  .WAIT  .CSIGEN .READW  .SETTOP .PRINT  .EXIT
7
8          ; LOADER - Load .SAV file into memory starting at location 0.
9          ;          Start program execution with interrupts disabled.
10         ;
11         ; To build LOADER, enter this program with your favorite editor
12         ; and type the following commands:
13         ;
14         ; .MACRO LOADER
15         ; .LINK LOADER
16         ;
17 000000   START:
18 000000   012703 000316'   MOV     #AREA+4, R3      ; a handy pointer
19
20         ; Claim memory up until RMON (or USR if LOCKed) for buffer.
21         ; Find first address after our partition.
22         ;
23 000004   .SETTOP #-2          ; R0 = last address
24 000012   005720   TST     (R0)+
25 000014   010002   MOV     R0, R2          ; R2 -> first loc after partition
26
27         ; Next, set command line and open input file on channel 3.
28         ;
29 000016   010601   5$:    MOV     SP, R1
30 000020   .CSIGEN #HILIM, #DEFEXT, #0
31 000034   010106   MOV     R1, SP
32 000036   010013   MOV     R0, (R3)      ; save high used address as buffer address
33 000040   .WAIT   #3          ; see if file specified
34 000046   103763   BCS    5$            ; and retry if not
35
36         ; Move 'move' routine to top of memory.
37         ;
38 000050   012701 000302'   MOV     #LAST, R1     ; R1 = first address after move routine
39 000054   014142   10$:    MOV     -(R1), -(R2) ; move a word
40 000056   001376   BNE    10$          ; until done
41
42         ; fill in rest of .READW ars block
43         ;
44 000060   005722   TST     (R2)+        ; R2 = address of routine
45 000062   010201   MOV     R2, R1      ; save routine address
46 000064   162302   SUB     (R3)+, R2   ; R2 = number of bytes left in partition
47 000066   000241   CLC
48 000070   006002   ROR     R2          ; R2 = number of words left
49 000072   010213   MOV     R2, (R3)   ; save in EMT ars block
50 000074   012700 000312'   MOV     #AREA, R0   ; R0 -> EMT block
51 000100   .READW
52 000106   103405   BCS    12$          ; read in the image
53 000110   014304   MOV     -(R3), R4   ; R4 -> image
54 000112   026401 000050   CMP     50(R4), R1  ; see if program HILIM is too high
55 000116   103005   BHIS   14$          ; if so, we probably didn't set whole image
56
57         ; to the move routine!

```

276

```

58
59 000120 000111 ; JMP (R1) ; to the routine
60
61 000122 12$: .PRINT #MESS ; come here on .READW failure
62 000130 .EXIT
63 000132 14$: .PRINT #OOPS ; come here if image too big
64 000140 .EXIT
65
66 .NLIST BIN
67 000142 MESS: .ASCIZ '?LOADER-F-Image read failed'
68 000176 OOPS: .ASCIZ '?LOADER-F-Image too big for memory'
69 .EVEN
70 .LIST BIN
71
72 ; The move routine (MUST BE PIC!)
73 ; In: R1 = first address of this routine
74 ; R4 = first address of image
75 ;
76 000242 000000 ; stopper
77 000244 012746 000340 MOV #340, -(SP)
78 000250 010746 MOV PC, -(SP)
79 000252 062716 000006 ADD #20$-., (SP)
80 000256 000002 RTI ; goto 20$ with interrupts off
81 000260 000005 20$: RESET ; stop the world
82 000262 005000 CLR R0
83 000264 012420 30$: MOV (R4)+, (R0)+ ; move image down to location 0
84 000266 020401 CMP R4, R1
85 000270 103775 BLO 30$
86
87 ; The next instruction sequence starts the image just read in.
88 ; .SAV files produced by RT-11 (for example FORTRAN programs with SIMRT)
89 ; are started by loading SP and PC from locations 42 and 40, respectively.
90 ; Alternately, an image that expects to start at location 0 may be started
91 ; by clearing the PC. Alternately, an image that expects to be started
92 ; through the power up vectors may be started by simulating a power-up.
93 ;
94 ; CLR PC ; the image expects to start at location 0
95 ; *** or ***
96 000272 013706 000042 MOV @#42, SP ; the image is a .SAV file
97 000276 013707 000040 MOV @#40, PC
98 ; *** or ***
99 ; MOV #24, SP ; the image comes up through the power-up
100 ; RTI ; vector
101
102 000302 LAST:
103 000302 073376 DEFEXT: .RAD50 /SAV/
104 000304 000000 000000 000000 .WORD 0, 0, 0
105 000312 004003 000000 000000 AREA: .WORD 10*400+3, 0, 0, 0, 0
106 000320 000000 000000
107 000324 HILIM:
108
109 000000' .END START

```

277



LOADER ID V03.02B13-MAY-80 16:59:32 PAGE 1-2  
SYMBOL TABLE

AREA	000312R	HILIM	000324R	MESS	000142R	START	000000R	...V2 = 000001
DEFEXT	000302R	LAST	000302R	OOPS	000176R	...V1 = 000003		

. ABS. 000000 000  
000324 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 1756 WORDS ( 7 PAGES)  
DYNAMIC MEMORY AVAILABLE FOR 49 PAGES  
LOADER,LOADER=LOADER

<b>μnote</b>		<b>NUMBER</b> 090
<b>TITLE</b> <u>USING THE BA11-VA IN SMALL SYSTEMS</u>		<b>DATE</b> 4 / 17 / 80
<b>DISTRIBUTION</b> <u>UNRESTRICTED</u>		<b>PRODUCT</b> BA11-VA MXV11-A, MXV11-A2
<b>ORIGINATOR</b> <u>RICK PLUMMER</u>		<b>PAGE 1 OF 2</b> <i>RBP</i>

There are a number of possible ways to configure small LSI-11 based systems. The purpose of this note is to point out some of the configurations and stimulate the imaginative solution of others. The configurations below will all be housed in a BA11-VA mounting chassis. This is ideally suited for ROM based, RAM based or TU58 mass storage based applications. Since the chassis contains no power sequencing, the CPU wake-up circuit will be used to initialize the system. Refer to Micro Note #85 for a further discussion of the wake-up circuitry. Remote restart may be accomplished with the connector provided. This note will also refer heavily to the MXV11-A multifunction module. Refer to the Microcomputer Processor Handbook and Micro Note #54 for more details on the MXV11-A.

1. Small ROM Based Systems

<u>Option</u>	<u>Modules</u>	<u>Functionality</u>
KD11-GC	KD11-HA	LSI-11/2 CPU
	MXV11-AC	8KB ROM 32KB RAM 2 Serial Ports

Customer program would be in PROM chips residing on the MXV11. Two option slots remain for other I/O devices.

2. Large ROM Based Systems

<u>Option</u>	<u>Modules</u>	<u>Functionality</u>
KD11-GC	KD11-HA	LSI-11/2
	MXV11-AC	32KB RAM 2 Serial Ports
MRV11-C	MRV11-C	64KB ROM

The MRV11-C may be addressed either directly or through a window. If window mapping is used, the MXV11-A2 bootstrap chip set may be used to load a program stored in PROM into the RAM memory for execution. One option slot remains for I/O modules.

**digital**  
**MICROCOMPUTER  
PRODUCTS  
GROUP**

### 3. RAM Based Systems


<u>Option</u>	<u>Module</u>	<u>Functionality</u>
KD11-GC	KD11-HA	LSI-11/2 CPU
	MXV11-AC	32KB RAM 2 Serial Ports Custom Boot

A typical RAM based system may be down-line loaded from a host processor. The customer down-line load boot may reside on the MXV11-A module. This system leaves two option slots for I/O. If memory expansion is desired, either an MXV11-AC or MSV11-DC may be added depending on the usefulness of two extra serial ports.

### 4. TU58 Based Systems

<u>Option</u>	<u>Module</u>	<u>Functionality</u>
KD11-GC	KD11-HA	LSI-11/2 CPU
	MXV11-AC	32KB RAM 2 Serial Ports 60 HZ Clock
MXV11-A2		Boot PROM
MXV11-AC		32KB RAM 2 Serial Ports

The TU58 based system is typical of the large amount of computer power that will fit in small spaces. Full system memory, bootstrap and four serial I/O ports occupy only three modules, leaving the forth for additional I/O capability.

	NUMBER 091
	DATE 4 / 29 / 80
	PRODUCT MRV11-C, MXV11-A2
	PAGE 1 OF 2 <i>RBP</i>

The purpose of this note is to provide users with the information to allow them to use the MRV11-C to bootstrap their system using standard MXV11-A2 bootstrap chips. While this is not usually optimum for new designs, due to the higher functional density of the MXV11-A Multifunction Module, it will be useful in upgrade and evaluation situations.

#### The MXV11-A2

There are two distinctly separate boot programs residing in a pair of 512 x 8 ROM chips. Address bit 8, the highest order bit, of the chip is used to select the appropriate one. One program boots from the TU58 while the other boots from RL02, RL01, RK05, RX02, RX01 or MRV11-C. The second will check system resources in the order listed; looking for a bootable device. See Micronote 62A for more detailed information concerning the bootstrap programs.

#### BOOTING FROM TU58

To use the TU58 boot simply:

1. insert PROM 040D1 in socket XE44
2. insert PROM 039D1 in socket XE43
3. jumper J117 to J112 to J107
4. jumper J116 to J111
5. jumper J 86 to J 87
6. jumper J 6 to J 7
7. jumper J 18 to J 19 to J20 to J21 to J22 to J24

The module will now have the TU58 boot starting at 173000.

#### BOOTING FROM DISK

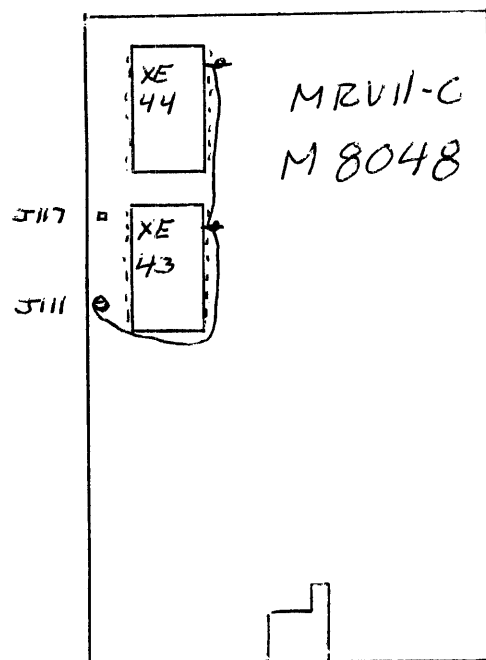
To use the Disk boot it is necessary to:

1. jumper the module as outlined in steps 3 thru 7 above.
2. Before inserting the ROM chips in the sockets, bend pin 23 of each of the chips upward 90 degrees. (This will void the chip warranty.) Now wire wrap a connection from pin 23 of 040D1 to pin 23 of 039D1. Wrap a second piece of wire to pin 23 of 039D1 and leave the other end free.

**digital**  
**MICROCOMPUTER**  
**PRODUCTS**  
**GROUP**

Now insert the ROM into the sockets as in steps 1 and 2 above. The free end of the second wire should now be wrapped to J111 for Disk boot or to J117 for TU58 boot.

If this is done carefully, no damage should be done to the chips.



<b>ynote</b>		<b>NUMBER</b> 092
<b>TITLE</b>	<u>TWO POTENTIAL PROBLEMS WITH THE RXV21 INTERFACE</u> (M8029)	<b>DATE</b> 5 / 8 / 80
<b>DISTRIBUTION</b>	<u>UNRESTRICTED</u>	<b>PRODUCT</b> RXV21
<b>ORIGINATOR</b>	<u>BARBARA BECK</u>	<b>PAGE</b> 1 OF 1 <i>RBP</i>


The RXV21 interface module may exhibit signs of irrational behavior due to two problems recently discovered with that board.

First, there exists a semiconductor-related problem which may cause undetected errors during data transfer. This problem will be detected by the RXV21 performance exercisor (CZRADA). Modules that are currently being shipped have been corrected for this. To identify modules that have already had the correction applied, it is necessary to check the revision level of the module. The revision level is stamped into the side of the handle and is identified by a letter. The old revision was "D" and this correction has updated the revision to "E". Another visual check to verify that the correction has been installed is to see if green wires have been added to chip E42.

Second, the Bus Address Register incorrectly increments over 32K word boundaries during DMA transfers. The implication here is that it will not work correctly in systems that have 11/23 processors that are running with memory management enabled. 11/23 development systems with RXV21 disks that are currently being shipped have been modified to correct this problem. They can be identified by the revision level -"F". For those customers that have upgraded their LSI-11's to 11/23's and need to have their RX02 floppy modified, please contact your sales person.

Please contact the LSI-11 Hot Line (800-225-9220) if you have any questions regarding these issues.

**digital**  
**MICROCOMPUTER**  
**PRODUCTS**  
**GROUP**

	NUMBER
	093
	DATE
	5 / 12 / 80
TITLE <u>USER WRITTEN SYSTEM TASKS UNDER RT-11 V4</u>	PRODUCT
DISTRIBUTION <u>UNRESTRICTED</u>	RT-11 V4
ORIGINATOR <u>JON TAYLOR</u>	PAGE 1 OF 1
	<i>EBP</i>

A SYSGEN option called "system tasking" is available in the RT-11 Version 4 foreground/background monitor. System tasking allows RT-11 to run up to six programs (called system tasks) in addition to the standard background and/or foreground programs. Although only Digital supplied system tasks are supported, it is not difficult to write programs which can be run as system tasks. This capability gives RT-11 true multi-tasking capabilities and allows it to be used when a small, fast, multi-tasking operating system is required.

Note that although RT-11 V4 has multi-tasking features, it remains a single user operating system - only one terminal at a time is the "console", with the power to run program development software. User written application tasks, however, may be assigned a terminal other than the console.

Be warned that any suspected system errors must be reproduced on a supported software configuration before submitting an SPR. This may be as simple as re-running the user-written system task as a foreground job to see if the problem persists. Details regarding the implementation of user written system tasks can be found in the RT-11 Software Support Manual Chapter 3, beginning on page 35.

**digital**  
**MICROCOMPUTER**  
**PRODUCTS**  
**GROUP**

<b>ynote</b>		<b>NUMBER</b> 094A
<b>TITLE</b> _____	RL02 SUPPORT BY DIGITAL SOFTWARE	<b>DATE</b> 5 / 13 / 80
<b>DISTRIBUTION</b> _____	UNRESTRICTED	<b>PRODUCT</b> RL02
<b>ORIGINATOR</b> _____	JON TAYLOR	<b>PAGE</b> 1 OF 1 <i>EBP</i>

This article describes the level of RL02 support offered by DIGITAL operating systems RT-11 Version 3B, RT-11 Version 4, and RSX-11M Version 3.2. Any applicable mandatory patches will be described. Patches are found in the "SOFTWARE DISPATCH" for the particular operating system.

#### RT-11 Version 3B

Although RT-11 V3B's "DL" handler was designed to interface to RL01 drives, RL02's are officially UNSUPPORTED under RT-11 V3B. In practice, they are reported to work on media with no bad blocks. If you must use RL02's (and/or RL01's) with V3B, review the following patches:

MONITOR	08M	Aug. 78
SOURCES	05M	Apr. 79
SOURCES	07M	Aug. 79
SOURCES	10M	Jan. 80
SOURCES	14M	Apr. 80
SYSTEM HANDLERS	01M	Sep. 78
SYSTEM HANDLERS	07M	Jan. 79
UTILITIES	25M	Apr. 80

#### RT-11 Version 4

The "DL" handler has been completely rewritten for V4. RL01 and RL02 drives are fully SUPPORTED. Version 4 distribution is available on RL01 and RL02 media. No special SYSGEN is required to use RL02 drives; RT-11 utilities will determine drive type "on the fly" using a .SPFUN call to the DL handler.

#### RSX-11M Version 3.2 and RSX-11S Version 2.2

RL02 support is available in 3.2 through SYSGEN. The system installer must explicitly tell the software how many RL01/RL02 drives are on an RLV11 controller and specify drive type (RL01 or RL02). After the ~~w~~irgin boot, an RL01 drive may be replaced with an RL02 (and vice versa). It is not necessary to re-SYSGEN in this case, just to re-boot. RL02 distribution kits for 3.2 were announced in Feb. 1980. The following mandatory patches are out:

2.1.6.1 M	Oct. 79
3.1.1.3 M	Apr. 80

**digital**  
**MICROCOMPUTER  
PRODUCTS  
GROUP**



<b>ynote</b>		<b>NUMBER</b> 095
<b>TITLE</b>	<u>VT103 APPLICATIONS FOR UNUSUAL BAUD RATES</u>	<b>DATE</b> 5 / 19 / 80
<b>DISTRIBUTION</b>	<u>UNRESTRICTED</u>	<b>PRODUCT</b> VT103
<b>ORIGINATOR</b>	<u>CHARLIE GIORGETTI</u>	<b>PAGE</b> 1 OF 1 <i>EBP</i>

Systems that require serial lines in addition to the console have configuration advantages in a VT103 system. The first additional serial line can be cabled internally and make use of the EIA connector on the rear of the VT103. Also, the serial lines that are added can be baud rate selectable from the SET-UP mode of the VT103. This feature allows easily changed baud rates and a wide range of speeds for the additional serial lines.

The VT103 in SET-UP mode B can select certain baud rate values. The baud rates that can be selected are 50, 75, 110, 134, 150, 200, 300, 600, 1200, 1800, 2000, 2400, 3600, 4800, 9600 and 19200. This wide range of values can be utilized with most DLV11 type interfaces and the MXV11. The baud rate of the VT103 console is fixed and is not affected by the given baud rate selected in SET-UP mode B for the additional serial lines.

The baud rate is selected using the TRANSMIT SPEED feature in SET-UP mode B on the VT103. The RECEIVE SPEED feature has no effect on serial line baud rates. If the serial line is a DLV11-J or a MXV11-AA, -AC configured for external baud rate and cabled into connector J2 on the Standard Terminal Port (STP) with DEC cable #19-11411-00, as described in the VT103 User's Manual Chapter 5, it would be able to be programmed from the keyboard in SET-UP mode B. If the line is a DLV11-E or a DLV11-F, set for external baud rate and common speed, cable the serial line into connector J3 on the STP with DEC cable BC08R-01. The baud rate is now selectable from the TRANSMIT feature.

<b>μnote</b>		<b>NUMBER</b> 096
<b>TITLE</b> <u>TU58 TIPS</u>		<b>DATE</b> 6 / 6 / 80
<b>DISTRIBUTION</b> <u>UNRESTRICTED</u>		<b>PRODUCT</b> TU58
<b>ORIGINATOR</b> <u>ERNIE STRANGE</u>		<b>PAGE</b> 1 OF 2

### KD11-F MEMORY REFRESH AND THE TU58

When the KD11-F goes through a memory refresh cycle, it takes approximately 130 usec. During the refresh cycle, no I/O device can be serviced. Operating the TU58 at 38.4K baud requires service every 260 usec. When reading or writing just to the TU58, no problems arise. If other devices are also serviced, depending on their number and type, the chances increase of not being able to service the TU58. Operating at a lower baud rate will lessen this potential problem.

### TU58 EMI

The drive units should be mounted in an area that is free from electro magnetic interference. Such noise could be caused by a switching power supply or a CRT. While wires coming from the read/write heads are shielded, these are low level signals and errors may occur if noise is introduced.

### TU58 HEAD CARE

The TU58 read/write head should be cleaned after the first 20 hours of use. Thereafter, cleaning should be done at 100 hour intervals. Use a cotton applicator moistened with DEC cleaning fluid (from cleaning kit TUC-01) or isopropyl alcohol, fluorocarbon TF or equivalent.

### TU58 CABLES

The BC20Z-25 cable should be used to connect the MXV11 or DLV11-J to the TU58. If the cable is being manufactured by the customer, pin 1 of the amp connector should not be used. The 38.4K baud clock on the MXV11 and DLV11-J will become loaded (Capacitive load) and cause the serial ports to malfunction. This problem usually occurs when 10 conductor mass terminated cable is used.

### TU58 POWER SUPPLY SELECTION

When considering a power supply for the TU58, insure peak and maximum current ratings are met.

**digital**  
**MICROCOMPUTER**  
**PRODUCTS**  
**GROUP**

## POWER CONSUMPTION

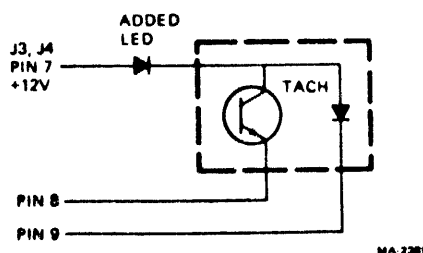
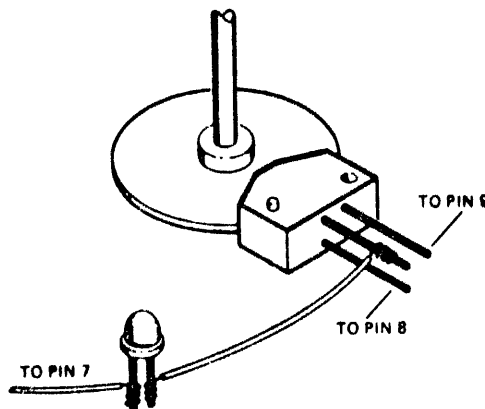
Logic Control Module  
and 1 or 2 Drives

11W, typical drive running  
+5V +5% at .75A, maximum  
+12V +10% -5% at 1.2A peak  
.6A average running  
.1A idle

If the power supply selected is rated at 1 amp for +12V, there may be a problem. The power supply could shut down when a peak current of 1.2A is drawn by the drive unit.

## RUN INDICATOR (TAPE MOVEMENT)

When the LED (Light Emitting Diode) is put in series with the tachometer, check for correct polarity. If correct polarity is not established, the tape could run off the reel because the logic card cannot read pulses from the tachometer, therefore, positioning is lost.

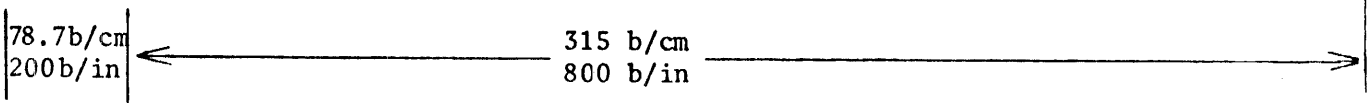


Installation of Run Indicator

**digital**  
**MICROCOMPUTER**  
**PRODUCTS**  
**GROUP**

<b>ynote</b>		NUMBER 097
TITLE	TU58 TAPE FORMAT AND ADDRESSING MODES	DATE 5 / 5 / 80
DISTRIBUTION	UNRESTRICTED	PRODUCT TU58
ORIGINATOR	ERNIE STRANGE	PAGE 1 OF 5

There has been some confusion on just how data was organized on the TU58 tape cartridge. The following information details actual tape formatting.



	Inter Record Mark 16-Bits	Header Sync. 16-Bits	Record Number 16-Bits	Compl. of Rcd. Number 16-Bits	Data Sync Field 56-Bits	Data Field 128 Bytes 1024-Bits	Check Sum 16-Bits	End Space All Zeros 136-Bits
TRACK 1	10 0.. 0	000..01	N + 1024	$\overline{N + 1024}$	00 00 00 00..01			
TR. 0		000..01	N	$\overline{N}$	00 00 00 00..01			

ONE RECORD

Header Description

The header and data fields are shown in the above figure. The header contains the following components.

- A. Interrecord Marks - Sixteen bits recorded at 200 BPI and having a data pattern of 1010101010101010. During search, the controller finds records by starting from a known position and counting the interrecord marks as they go by at 60 IPS. When it reaches the record before the record being searched for, it slows the tape down to 30 IPS and reads the next header. If the record number read agrees with the record number expected, and it error checks O.K., the controller continues with the read or write operation. Otherwise, it corrects the current position and initiates a new search.

Any combination of incorrect record numbers or tape reversals totaling eight events causes the controller to abort the operation and declare a seek error.

**digital**  
**MICROCOMPUTER**  
**PRODUCTS**  
**GROUP**

- B. Header Sync - (All of the following bits are recorded at 800 BPI.) It consists of 15 zeros followed by a one. The controller looks for the one and then begins to accept the record number.
- C. Record Number - 16-Bits (0 to 2047)
- D. Record Number Complement - The controller tests this number to insure that the header was read with no errors.
- E. Data Sync Field - 55 zeros and a one. During a write operation, the controller turns on write current after the first 8 zeros and writes the remaining zeros and one. The glitches caused by switching on the write current are then confined to a narrow space which the controller blanks out during read operations. After a fixed duration blank (controller ignores tape output) the controller begins to search for the one at the end of the data sync field. When it finds the one, it begins reading the data field.

## Data Field Description

- A. Data Field - The next 1024-bits of data are stored in the data buffer in the controller.
- B. Checksum - The checksum contains 16-bits and is used to find errors in the real data. During read, each pair of bytes is summed in a 16-bit add. If a carry results, it is added to the LSB (1's complement addition). If the sum of 128 bytes does not equal the checksum on tape, the record is re-read up to eight tries. If the correct data cannot be read after eight tries, a hard error is indicated to the host processor.
- C. Post Record Zeros - When the tape is formatted, an additional 136 zeros are written to allow for 10% tolerance in motor speed. The first 8 zeros are written after data is written. The remaining 128 zeros are never rewritten or read in normal operation. Their purpose is only to provide flux reversals in the space between the end of data and the beginning of the next record.

	Record 1024	Record 1025	Record 1026	Record 2046	Record 2047		
BOT	H DATA	M H DATA	M H DATA	M H DATA	M H DATA	EOT	TRACK 1
BOT	H DATA	M H DATA	M H DATA	M H DATA	M H DATA	EOT	TRACK 0
	Record 0	Record 1	Record 2	Record 3	Record 1022	Record 1023	

TAPE FORMAT

M = Interrecord Mark

H = Header

No Mark for Records 0, 1024

BOT, EOT, and Interrecord Marks

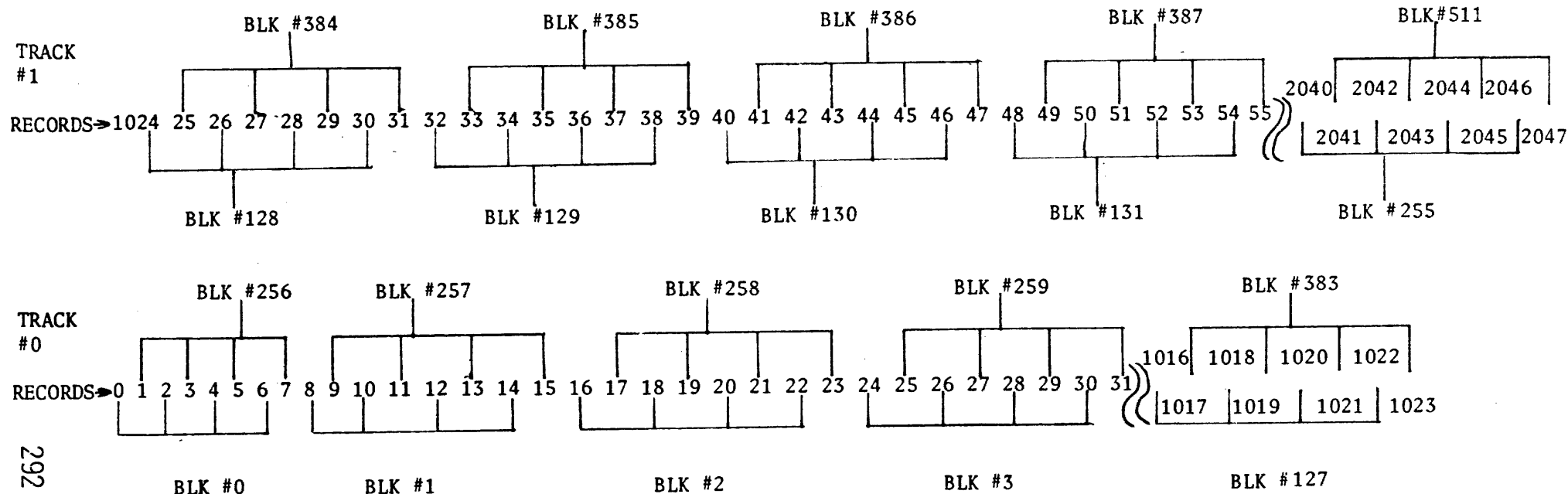
Special marks are recorded during tape formatting to indicate beginning and end of tape as well as beginning of record. These marks are recorded at one fourth the bit density of data or at 200 RPI. The lower frequency is detected by the controller. The encoding method used is not sensitive to tape speed, so that ones and zeros may be recovered at the lower frequency with no change in hardware. The BOT, EOT, and IRM (Interrecord Marks) are distinguished from one another as follows:

- A. BOT is recorded as all zeros.
- B. EOT is recorded as all ones.
- C. IRM's are recorded as alternate ones and zeros.

291

BLOCKS ARE INTERLEAVED TO IMPROVE ACCESS TIME

A



292

TU58 RECORD AND BLOCK DIAGRAM

Logical Format

Data is recorded on two tracks. Each track contains 1024 records of 128 bytes. To accommodate the orientation of the record and erase head gaps, both tracks are recorded in the same direction. To accommodate standard mass storage blocks of 512 bytes, the controller groups four 128 byte records together. Addressing from the host is done by 512 byte block numbers. (0-511)

1024 Data Bits Per Record  
128 Bytes Per Record  
4 Records Per Block  
256 Blocks Per Track  
2 Tracks Per Cartridge  
One Cartridge Equals 256K Bytes  
(262, 144 Bytes)

A special mode permits access of 128 byte data blocks. Addressing in special mode is by 128 byte block numbers (0-2047).

## ADDRESSING MODES

### Special Addressing Mode

Setting the most significant bit of the modifier byte (byte 3 of command packet) to 1 selects special address mode. In this mode all tape positioning operations are addressed by 128-byte blocks (0-2047) instead of 512-byte blocks (0-511). Zero-fill in a write operation only fills out to a 128-byte boundary in this mode. Applications that have less than 128-bytes of data per block could use the special addressing mode to put more data on a cartridge.

### Normal Addressing Mode

Tape motion stops at the end of a block in a write operation and at the end of a record in a read operation. Writing or reading data to the TU58 is done by specifying the block and number of bytes to be written or read. If only one record (128 bytes) is to be written, the remaining three records (384 bytes) of the block will be zero-filled.



<b>ynote</b>		<b>NUMBER</b> 098
TITLE <u>11/23 &amp; 11/03 RL01 BASED PACKAGED SYSTEM EXPANSION</u>		DATE 5 / 19 / 80
DISTRIBUTION	<u>UNRESTRICTED</u>	PRODUCT RL01 Based Packaged Development Systems
ORIGINATOR	<u>MIKE SHUSTER</u>	
		PAGE 1 OF 2

All 11/03 and 11/23 Packaged Systems have been described as expandable to a second BA11-NE or -NF type box within a system cabinet. The above statement is true but must be qualified to reflect power controller ratings in 11/03 and 11/23 systems with RL01 disk.

11/03 SYSTEMS

120V 11/03 PACKAGED SYSTEMS

These systems will continue to be produced with the 871-A 12 amp power controller and a standard 15 amp plug (Fig. 1).



FIGURE 1

120V 11/03 systems have the physical space for a BA11-NE expander box. It must be noted, however, that the current capacity to power an additional box plus mass storage does not comply with UL ratings.

IMPACT ON 11/03 USERS

11/03 users will be affected when adding a second BA11-NE or -NF box.

Should a user wish to expand an existing system, he or she must replace the 871-A 12 amp power controller with an 871-C 16 amp power controller, as well as provide a 20 amp receptacle, in order to comply with UL ratings.

All 240V 11/03 Packaged Systems are expandable.

## 11/23 SYSTEMS

### 120V 11/23 PACKAGED SYSTEMS

These systems are being produced with an 871-C 16 amp power controller which allows expansion to a second BA11-NE box. The 871-C has a plug requiring a 20 amp outlet (Fig. 2).



FIGURE 2


A few early production 11/23 packaged systems used the 871-A 12 amp power controller with a standard 15 amp plug. The procedure for expanding these systems is similar to that of the 120V 11/03 Packaged Systems.

### IMPACT ON 11/23 SYSTEMS

11/23 Packaged System users must understand that 120V systems require a 20 amp outlet which is standard in most commercial buildings. If a 20 amp outlet is unavailable, a licensed electrician should be consulted for the best possible solution.

All 240V 11/23 Packaged Systems are expandable.

**\*NOTE:** This analysis does not include terminals (a VT100 is rated for 3 amps at 120V). It is recommended the wall outlets be used for such a purpose so that the system will remain within its UL rating.

		NUMBER 099
		DATE 8 / 15 / 80
		PRODUCT BDV11
		PAGE 1 OF 1
TITLE	RL02 BOOTSTRAP FAILURES USING BDV11	
DISTRIBUTION	UNRESTRICTED	
ORIGINATOR	RICK PLUMMER	

A problem has been discovered when using the BDV11 to bootstrap RL02 disks. The symptom would be seemingly intermittent boot problems.

The problem is due to the fact that the BDV11 boot code was written for the RL01, which uses only eight cylinder address bits (versus nine for the RL02). When calculating the difference count for a seek to cylinder 0, bit 15 is cleared (it is defined as "must be 0" for RL01's). This means that if the heads are already positioned at cylinder 400, or beyond, an incorrect difference will be calculated. A seek will then be made to the incorrect cylinder, resulting in bootstrap failure. Since the position of the heads determines the failure mode, the problem will appear to be intermittent. In fact, many users may never see the problem, particularly those who only bootstrap after power-up.

The problem will be corrected with new ROM's. Until these ROM's are available, it may be necessary to avoid the error by cycling the desired RL02 down, then up again before bootstrapping.

Contact the LSI-11 hot line for further information.

<b>μnote</b>		<b>NUMBER</b> 100
<b>TITLE</b> <u>UPGRADES FOR PB11</u>		<b>DATE</b> 11 / 13 / 80
<b>DISTRIBUTION</b> <u>UNRESTRICTED</u>		<b>PRODUCT</b> PB11
<b>ORIGINATOR</b> <u>CHARLIE GIORGETTI</u>		<b>PAGE</b> 1 OF 2

User applications that require Intel 2732, 2732-6 or 2716 type EPROMs should consider the following upgrades to the PB11 hardware and the PROM/RT-11 PROM programming utility.

The first upgrade is required when 2732s or 2732-6s are used with PROM/RT-11 V1.05A under distributed RT-11 V03B or RT-11 V04 monitors. The upgrade is also required when 2716s are used with PROM/RT-11 V1.05A under a distributed V04 monitor. The reason for the upgrade is to increase a time out counter that is activated when data is being transferred to the PB11 hardware. The following patch to PROM.SAV or PROM.REL changes the counter and upgrades the version number:

```

.RUN PATCH

FILENAME-

*PROM.SAV/C

*15523;0R

*10226;1R

*0,5070\           101       102

*1,3070/           74        264

*E

Checksum?           1377

```

(underlined characters typed by the user)

The second upgrade is required when the 2732s are used. In the option, PB11K-AD, for 2732s, the socket adapter is 715-1531. The change requires the replacement of two resistors on the reverse of the socket adapter itself. The resistors are marked R1 and R2, both have a resistance of 1 KOhms. Resistor R1 should be changed to 10 KOhms and resistor R2 should be changed to 3.3 KOhms. Figure 1 shows the resistor locations.

**digital**  
**MICROCOMPUTER**  
**PRODUCTS**  
**GROUP**

The resistor changes can be implemented by the user, by a local DATA I/O service office, or by Digital's Customer Returns Area. The Customer Returns Area could reject the adapter for future repairs when this resistor change is implemented by non-authorized personnel.

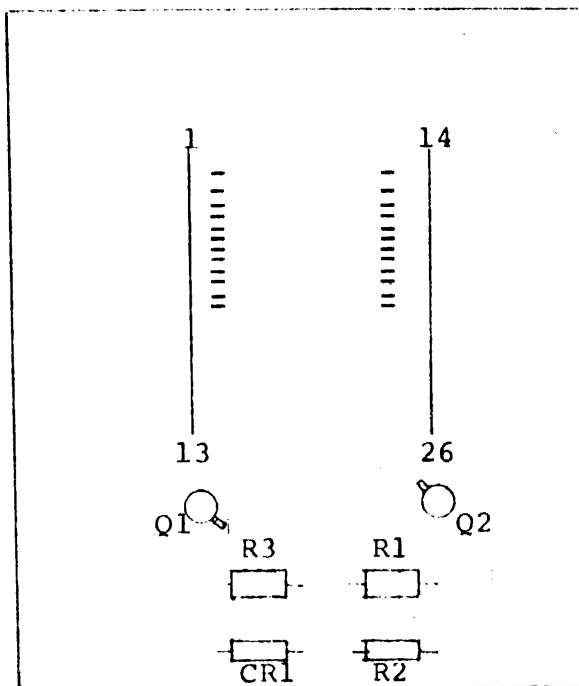


Figure 1 reverse of socket  
adapter 715-1531

<b>note</b>		<b>NUMBER</b> 101
<b>TITLE</b> <u>TU58 SYSTEM POWER PROBLEM</u>		<b>DATE</b> 1 / 26 / 81
<b>DISTRIBUTION</b> <u>UNRESTRICTED</u>		<b>PRODUCT</b> TU58
<b>ORIGINATOR</b> <u>ERNIE STRANGE</u>		<b>PAGE</b> 1 OF 1

### SYSTEMS

When DLV11-J's were used in a system to communicate with TU58's, an extra continue response from the TU58 was obtained. The DLV11-J's -12V charge pump was coming up late. Initially, the TU58 logic card would start its self-test diagnostics. Then the TU58 received what it thought was a break (-12V coming up on the DLV11-J), creating a continue response from the TU58. The TU58 logic card, ECO #2 (CS Rev. C to C1) corrected this problem.

### TU58 POWER SUPPLY SELECTION

When considering a power supply for the TU58, insure peak and maximum current ratings are met.

### POWER CONSUMPTION

Module and 1 or 2 drives	11W, typical, drive running
	+5V +5% at .75A, maximum
	+12V +10% -5% at 1.2A, peak
	.6A average running
	.1A idle

If the power supply selected is rated at 1 amp for +12V, there is a problem. The power supply could shut down when a peak current of 1.2A is drawn by the drive unit.

<b>μnote</b>		<b>NUMBER</b> 102
<b>TITLE</b> <u>MMU CONFIGURATION JUMPERS</u>		<b>DATE</b> 1 / 14 / 81
<b>DISTRIBUTION</b> <u>LSI-11/23 USERS</u>		<b>PRODUCT</b> KDF11 (M8186)
<b>ORIGINATOR</b> <u>LINDA MENTZER</u>		<b>PAGE</b> 1 OF 1

The purpose of this Micronote is to provide the proper configuration for the KDF11-A module to be used when an MMU chip is present.

FLOATING POINT COMPATIBLE MMU. DIP PART #21-15542-01 (CHIP PART #304E)

Etch Revision C modules should have jumper W2 removed and W3 installed when the above MMU chip is present.

Etch Revision A modules must have ECO M8186-ML009 installed if the above MMU chip is used. This ECO is to remove W2 and install a wire from E2 pin 5 to E2 pin 15.

If the proper configuration is not followed, a timing problem could occur which would cause the MMU chip to place the wrong physical address on the bus. The frequency of occurrence of the problem would increase with increasing temperature of the chip or with installation of the Floating Point option.

NON-FLOATING POINT COMPATIBLE MMU. DIP PART #21-15542-00 (CHIP PART #304C)

Etch Revision C modules should have W2 inserted and W3 removed.

Etch Revision A modules should have W2 installed.

NEED MORE INFORMATION?

Please contact the LSI-11 Hot Line (800-225-9220) if you have any questions regarding this.

<b>ynote</b>		<b>NUMBER</b> 103
<b>TITLE</b> <u>CREATING A DIAGNOSTIC DECTAPE II UNDER XXDP+</u>		<b>DATE</b> 9 / 30 / 80
<b>DISTRIBUTION</b> <u>UNRESTRICTED</u>		<b>PRODUCT</b> TU58
<b>ORIGINATOR</b> <u>BOB DUCEY</u>		<b>PAGE 1 OF 1</b>

The following procedure will create an XXDP+ diagnostic monitor on a TU58 tape. Original XXDP+ diagnostic release media is used as the source and various commands from UPD2 do the transfer to the destination device. The resulting copy will be a DDDP+ diagnostic tape.

- 1/ Boot XXDP+ from the distribution media.
- 2/ Answer the date, 50 HZ and LSI questions.
- 3/ Run UPD2.  
eg. R UPD2
- 4/ Take a scratch DECTAPE II and load into the TU58 drive 0.
- 5/ Use the zero command, this will initialize the tape.  
eg. Zero DDO:(CR)
- 6/ Use the load command to load the TU58 monitor (HMDDAX.SYS) into a buffer within the utility program.  
eg. Load DEV:HMDDAX.SYS (where X is the current version of the monitor 0,1 etc.)
- 7/ Use the dump command to write the contents of buffer area in the utility to the TU58.  
eg. Dump DDO:HMDDAX.SYS
- 8/ Use the SAVM command to create the boot blocks on the tape.  
eg. SAVM DDO:HMDDAX.SYS
- 9/ Use the PIP command to copy all desired files from the system device to tape.  
eg. PIP DDO:FILNAM.TYP=DEV:FILNAM.TYP.  
(Where DEV: is the source device.)
- 10/ You now have a bootable TU58 diagnostic tape. If you wish to exercise the TU58, the TU58 performance exerciser is diagnostic ZTUUBX.BIN. (Where X is the current version level 0,1 etc.)



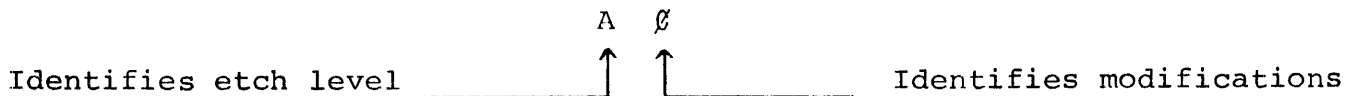
<b>uNote</b>		<b>NUMBER</b> 104
TITLE	11/23 ECO Status	<b>DATE</b> 12 / 3 / 80
DISTRIBUTION	Unrestricted	<b>PRODUCT</b>
ORIGINATOR	Jon Taylor	<b>PAGE 1 OF 7</b>

This uNote documents the ECO and etch revision history of the KDF11-A module. A quick verify has been included so that the status of a module may be determined by a visual check.

In using the attached table, it is important to be aware of the new hardware revision system employed on the M8186 (KDF11-A). Whereas other modules have the etch revision level coded into the etch and the circuit schematic (cs) revision level on the module handle, the M8186 has both stamped on the module handle.

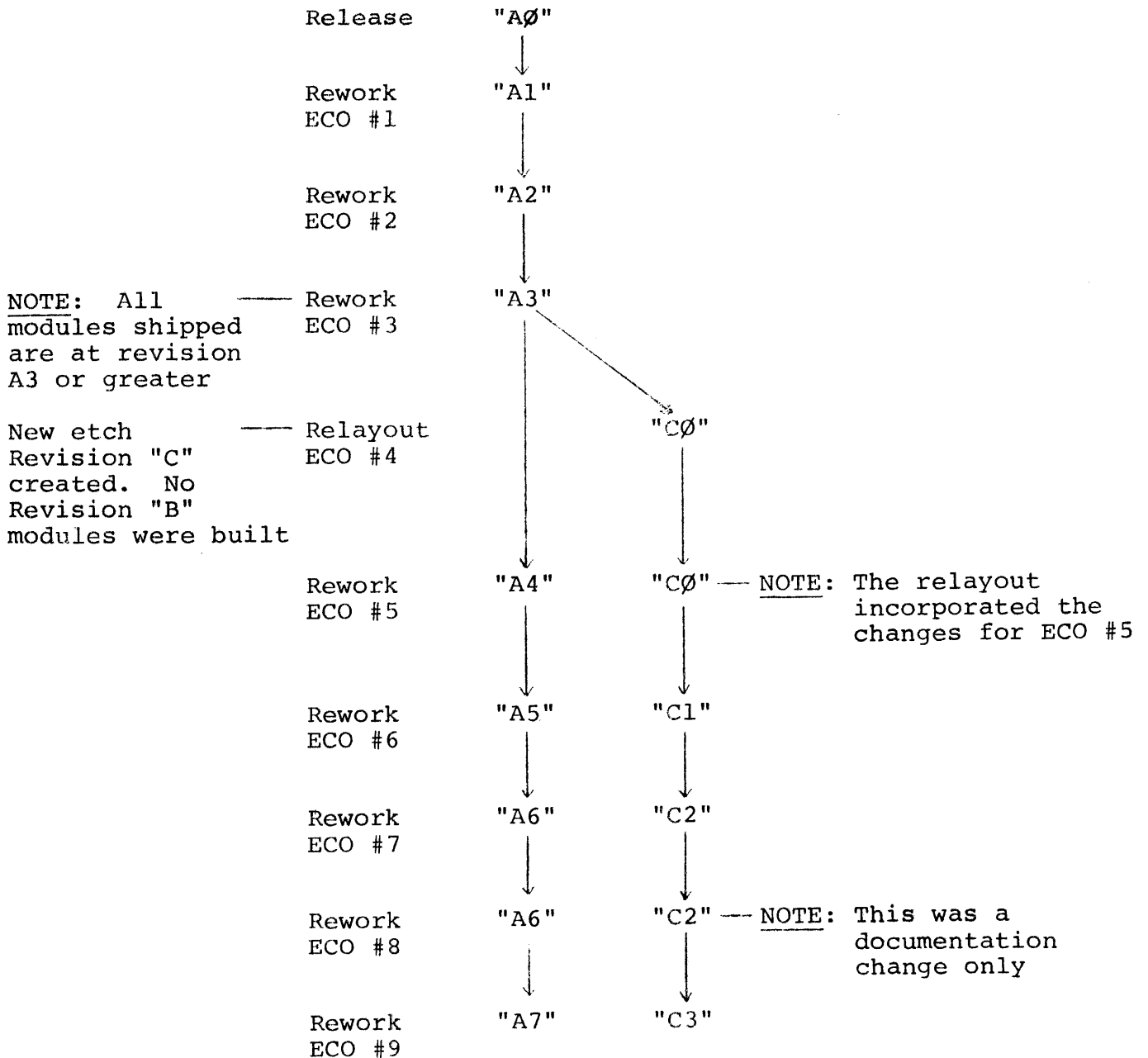
The revision identifier is a two-field alphanumeric designation. The first field indicates the etch revision. The second field indicates the modifications ("cs revision") to this etch.

Hardware revision notation:



The M8186 began as hardware revision "AØ", as shown above. That is, etch revision "A" with no modifications or rework. As ECO's were released calling for rework (but not a new etch), the hardware revision level was updated to "A1", then "A2", etc. When new etch revisions were released, the etch revision field was incremented from "A" to "B" to "C". (For the M8186, no etch revision "B" modules were built, so the revision level appeared to change from "A" to "C" via ECO #4. Etch Revision "C" boards had ECO #5 incorporated in them before release, therefore, their hardware revision status did not change with ECO #5).

The hardware revision history of the M8186 is shown below:



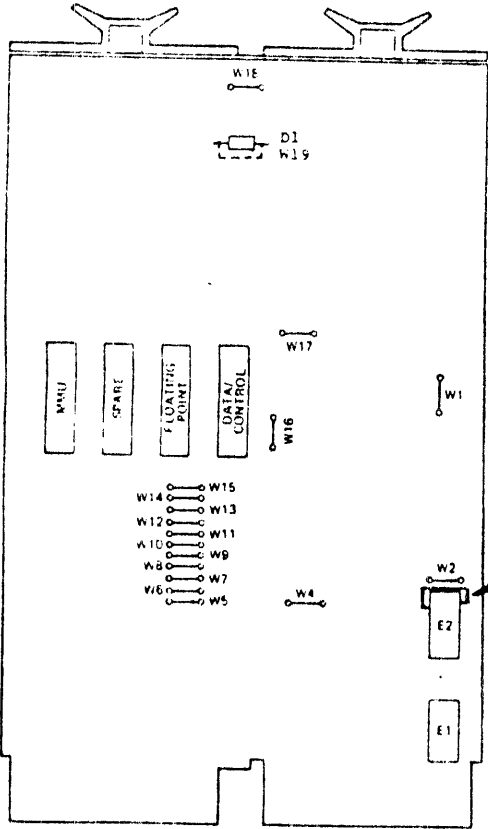
JUMPER FUNCTIONS ON ETCH REVISION "A" AND "C" MODULES

	JUMPER	FUNCTION	DESCRIPTION	SHIPPING STATUS														
E T C H " A " A N D E T C H " C "	W1	Master Clock	In = Enabled, Do Not Remove	In														
	W2	Reserved for DEC	Factory Configured, Do Not Alter	Out														
	W3*	Reserved for DEC	Factory Configured, Do Not Alter	In														
	W4	Event Line Enable	Out = Enabled	In														
	W5, W6	Power-Up Mode	<table border="1"> <thead> <tr> <th>Mode Name</th> <th>W5</th> <th>W6</th> </tr> </thead> <tbody> <tr> <td>0   PC@24, PS@26</td> <td>Out</td> <td>Out</td> </tr> <tr> <td>1   Console <math>\mu</math>ODT</td> <td>In</td> <td>Out</td> </tr> <tr> <td>2   Bootstrap</td> <td>Out</td> <td>In</td> </tr> <tr> <td>3   Extended <math>\mu</math>Code</td> <td>In</td> <td>In</td> </tr> </tbody> </table>	Mode Name	W5	W6	0   PC@24, PS@26	Out	Out	1   Console $\mu$ ODT	In	Out	2   Bootstrap	Out	In	3   Extended $\mu$ Code	In	In
Mode Name	W5	W6																
0   PC@24, PS@26	Out	Out																
1   Console $\mu$ ODT	In	Out																
2   Bootstrap	Out	In																
3   Extended $\mu$ Code	In	In																
W7	Halt/Trap Option	In=Trap to I/O on "Halt" Out=Enter console $\mu$ ODT on "Halt"	Out															
W8	Bootstrap Address	In=773000 is bootstrap address Out=Bootstrap address is specified by Jumpers W9-W15	In															
W9-W15	User Selectable Bootstrap Address	W9-W15 Correspond to Address Bits 9-15 respectively. In=Logic "1" Out=Logic "0"	In															
W16, W17	Reserved for DEC	Factory Installed, Do Not Remove	In															
Etch "C"	W18	Wake-Up Circuit	Out = Enabled	In														
Etch "A"	W18	Reserved for DEC	Factory Installed, Do Not Remove	In														
	W19	Wake-Up Circuit	Out = Enabled	In														

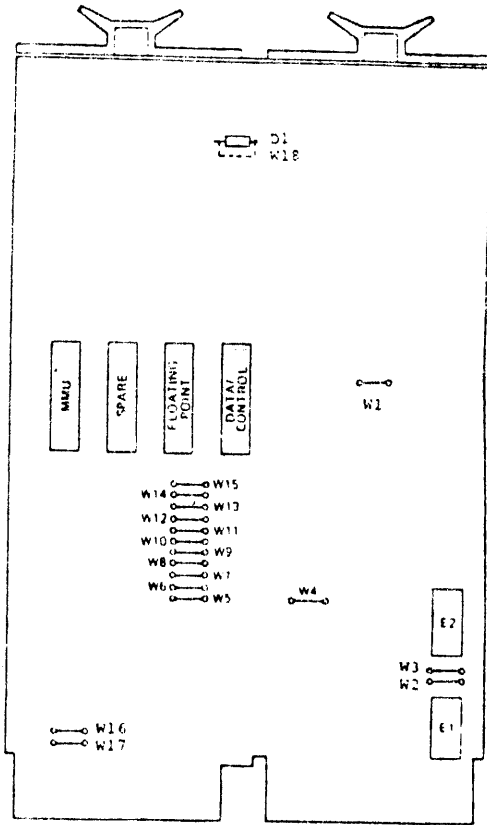
\*W3 on Etch Revision "A" Modules consists of a jumper from E2 Pin 5 to E2 Pin 15.

**digital**

**MICROCOMPUTER  
PRODUCTS  
GROUP**



M8186 ETCH REVISION "A"



M8186 ETCH REVISION "C"

LOCATION OF JUMPERS ON ETCH REVISION "A" AND REVISION "C" MODULES

The following table details the ECO's issued since the M8186 began to ship to the field. These ECO's are coded "M8186-ML00X", where "X" is the ECO shown below:

ECO #	PROBLEM	QUICK VERIFY
4	Too many wires and etch cuts, new etch needed. Note that the jumper locations change for etch Revision "C".	Module handle will be stamped "Cn" (Where "n" refers to the CS revision.)
5	I/O page addressing scheme differs from LSI-11/2 processor. <u>Note: Implementation of this ECO impacts configurations. This ECO is discussed in detail in uNote 80A.</u>	Check for etch cut to E7 Pins 16 and 18
6	The internal wake-up circuit defeats the sequencing provided by standard DEC power supplies. This ECO should be installed when the M8186 is used with same.	Red jumper wire is installed in parallel with D1.
7	CTL/DAT hybrid (57-00000-00) and MMU IC (21-15542-00) were not compatible with KEF11-AA floating point option. The FP registers in the MMU were inaccessible, and the CTL/DAT data path caused intermittent errors in floating point instructions.	CTL/DAT should be 57-00000-01 and MMU should be 21-15542-01 for floating point compatibility. <u>Coordinate with ECO M8186-ML009.</u>
8	MMU (21-15542-01) was included as part of the M8186 module. It should have been specified as an option that could be added to the module. This ECO removes the MMU from the M8186. ECO KDF11A-ML001 adds it back in at the option level (KDF11-AB, KDF11-AA). This is a documentation change only.	Modules in systems may or may not have MMU, depending upon which option they represent. Spares, however, are ordered as modules and will <u>not</u> have MMU's. MMU's must be ordered separately.
9	1. No jumper table in printset (documentation change only)  2. Crystal oscillator may short to adjacent components.	1. Prints contain a jumper table  2. Oscillator has nylon spacer. Manufacturing change only.

ECO # PROBLEM

QUICK VERIFY

- 9      3. Possibility of worst case MMU timing violations. Change configuration of W2 and W3 to adjust timing. This ECO must be installed:
- A) When ECO M8186-ML007 is put in
  - B) When a ucode option (i.e. KEF11) is installed.
  - C) When a 40 Pin IC (CTL/DAT, MMU or KEF11) is replaced.
  - D) Whenever unexplained system crashes are occurring.
3. Module will have W2 removed and W3 in. On etch Rev "A" modules, W3 is installed by soldering a jumper wire from E2 Pin 5 to E2 Pin 15.

ECO's in Other Options, but Related to the 11/23

- RXV21 (M8029) - Modules at CS Revision "E" and lower cannot correctly DMA across a 64Kb boundary. This will cause DECX to fail in an 11/23 and could affect custom or modified device drivers. It does not affect modules installed in 11/03-based systems. Modules installed in 11/23's must be at CS Revision "F" or higher.
- BDV11 (M8012) - Prior to ECO #2, the BDV11 did not provide proper termination for BIRQ6. Modules installed on 11/23's should be at CS Revision "D" or higher. (Only a few early units are affected.)
- DLV11-J (M8043)- The DLV11-J will respond to incorrect bus addresses when used on 11/23 CPU's unless ECO #2 is installed. Modules used on 11/23 systems must be at CS Revision "E" or higher.

<h1>μnote</h1>	NUMBER
	105
	DATE
	2 / 3 / 81
TITLE <u>MXV11 BOOTSTRAP PROBLEMS</u>	PRODUCT
DISTRIBUTION <u>UNRESTRICTED</u>	MXV11
ORIGINATOR <u>BEVERLY KAY</u>	PAGE 1 OF 2

Two problems have been noted at bootstrap time in systems containing an LSI 11/23 processor and an MXV11.

The first problem occurs when using the MXV11-A2 ROMs to boot from a TU58. The bootstrap program halts when executed immediately after power-up, in processor power-up mode 1 or 2. However, when the boot is tried again from ODT (773000G) it is successful.

The cause of this behavior lies in the TU58 bootstrap code on the MXV11 ROMs (refer to unote 62A). The MFPS instruction at location 173024 moves only the low byte of the processor status word into location 6. Memory is sized by writing successively to each location until a non-existent location causes a trap to 4. The new PSW is picked up from the word at location 6, but only the low byte has been initialized. The high byte contains random data.

Bits 14 and 15 of the PSW determine the mode of the processor, kernel or user. If location 6 has 1's in these bits, the processor will go into user mode causing some differences and restrictions in operation. For example, a HALT instruction causes a trap to 10, a RESET is executed as a NOP, and different stack pointers are used in each mode.

The ROM bootstrap completes execution and the secondary bootstrap read from the TU58 begins. The secondary boot executes until it encounters an illegal instruction - i.e., a HALT. The failure mode will differ depending on the secondary boot read from the tape. The system will boot successfully the second time it is attempted because a subsequent routine in the ROM code, MEMT, writes 0's into memory. When the boot is tried again, the upper byte has been cleared, and the status is picked up as intended - in kernel mode.



Note that this problem will occur only in 11/23/MXV11/TU58 systems, and the boot will succeed if location 6 is cleared before starting execution. This error has been corrected in a new version of the MXV11-A2 ROM's.

The second problem is evidenced by the processor hanging on power-up. By halting the processor (break key, halt switch) and going into ODT, then typing 773000G, or pushing the restart switch, the system will boot. For this second problem, the bootstrap device is not limited to the TU58, and the ROM code is not necessarily the MXV11-A2 code.

This problem is caused by a failure to initialize a latch on the MXV11 at power-up. This latch enables the gating of DIN, and subsequently generates BRPLY. The result is a BRPLY may be sent by the MXV11 even though it has not been addressed. ECO M8047-MR002 corrects this problem. Installation of this ECO brings the MXV11 to circuit schematic revision C.

If you have any further questions, please contact the LSI-11 Hotline.



TITLE MXV11 FUNCTIONALITY  
DISTRIBUTION UNRESTRICTED  
ORIGINATOR Bruce Gollob

NUMBER  
106

DATE  
4 / 29 / 81

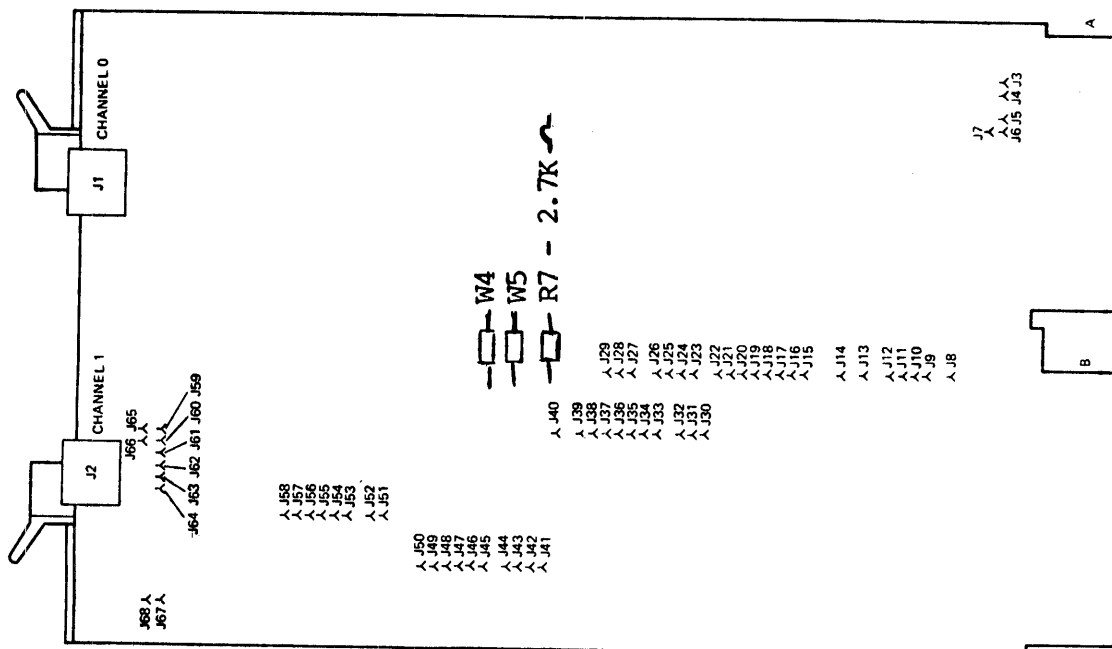
PRODUCT  
MXV11

PAGE 1 OF 3

The purpose of this u-note is to describe the functionality of the MXV11-A series modules, and describe a system configuration with two MXV11-AC multifunction modules.

### MXV11-A FUNCTIONALITY

The multifunction module (MXV11-A series) contains two asynchronous line interfaces, space for two user configured ROMs or the system device bootstrap ROMs, memory with on board refresh, and a crystal clock which can be used to generate the system line time clock. There is no provision for battery backup, or to allow addressing of the top 2K words of memory in a 30K word system. The board has 18 bit address decoding, however, the RAM memory decodes only 16 bits, therefore in a system with more than 32K words of memory, the MXV11-A on board memory must be placed in the range of 0-32K words (000000-177776). Removal of the zero ohm resistor W4 on the MXV11-AA or W5 on the MXV11-AC will totally disable the RAM memory on the multifunction module.



**MICROCOMPUTER  
PRODUCTS  
GROUP**

## SYSTEM CONFIGURATION WITH TWO MXV11-AC MODULES

When a system configuration requires multiple MXV11-AC modules, special care must be taken to insure that no two options on the boards have the same address. In the system configuration shown below there is no memory overlap and each slv address and vector is unique. It is only necessary to have one bootstrap and line time clock per system. Therefore, the bootstrap and the line time clock on the other board must be disabled. Channel 1 on any MXV11-A series module, whether it is the console or not, can be configured to ignore break, to halt on break or reboot on break. To disable an option on the multifunction module, all of the jumpers associated with the option must be removed. This must be done, otherwise the board is left in an unknown state which could cause unpredictable results.

The chart shown below shows one way to configure two MXV11-AC modules in a system. The system configured contains 32K words of RAM, the MXV11-A2 disk bootstrap, and four asynchronous serial line interfaces, one of which is the console interface.

				MXV11-AC #1		MXV11-AC #2		
				from	to	from	to	
-----								
RAM				J30*	J31	J30*	J31	
#1 BANK 0-3 (000000-077776)				J32*	J33	J31	J33	
#2 BANK 4-7 (100000-160000)				J31*	J32	J32	J34	
-----								
SLU ADDRESSES				J23*	J18	CH. 0	J23*	J18
				J24*	J19		J24	J12
				-----				
				J26*	J15		J26	J16
CH. 0	#1	#2		J25*	J14	CH. 1	J25	J17
CH. 1	176500	176510		J27*	J13		J27*	J13
	177560	176520		J28*	J19		J28*	J19
-----								
SLU VECTORS				J56*	J51		J56	J58
				J54*	J52		J55	J57
CH. 0	#1	#2		J53*	J54		J54*	J52
CH. 1	300	310		J53*	J57		J53	J51
				-----				
ROM BOOT (DISK)				J37*	J38		J8	J21
				J21*	J22		-	-
ROM ON #2 DISABLED				J34*	J37		-	-
				J33*	J39		-	-
				J29	J16		-	-
-----								

		MXV11-AC #1		MXV11-AC #2	
		from	to	from	to
SLU PARAMETERS	8 DATA BITS	J59*	J61	J59*	J61
	EVEN PARITY	J61*	J62	J61*	J62
	1 STOP BIT	J60*	J63	J60*	J63
	8 DATA BITS	J62*	J64	J62*	J64
	EVEN PARITY	J64*	J66 CH.0	J64*	J66
	1 STOP BIT	J63*	J65	J63*	J65
BAUD RATE	9.6K	J46*	J48 CH.1	J46*	J48
	9.6K	J45	J48 CH.0	J45	J48
BREAK GENERATOR (NOT USED, REMOVE FACTORY JUMPERS)					
CRYSTAL CLOCK		J68*	J67	J68*	J67
LINE TIME CLOCK		J3*	J4	-	-

\*FACTORY INSTALLED JUMPERS.

<b>uNote</b>		NUMBER 107
TITLE	<u>22-BIT ADDRESSING FOR DMA CHIPKIT USERS</u>	DATE 07 / 16 / 81
DISTRIBUTION	<u>Unrestricted</u>	PRODUCT CHIPKIT
ORIGINATOR	<u>Chris DeMers</u>	PAGE 1 OF 2

This uNote is intended to assist chipkit users who are designing DMA interfaces for use with 22-bit (Q-22) addressing in LSI-11/23 based systems.

To extend the DMA addressing to 22 bits, the following must be accomplished in addition to using the chips provided in the chipkit:

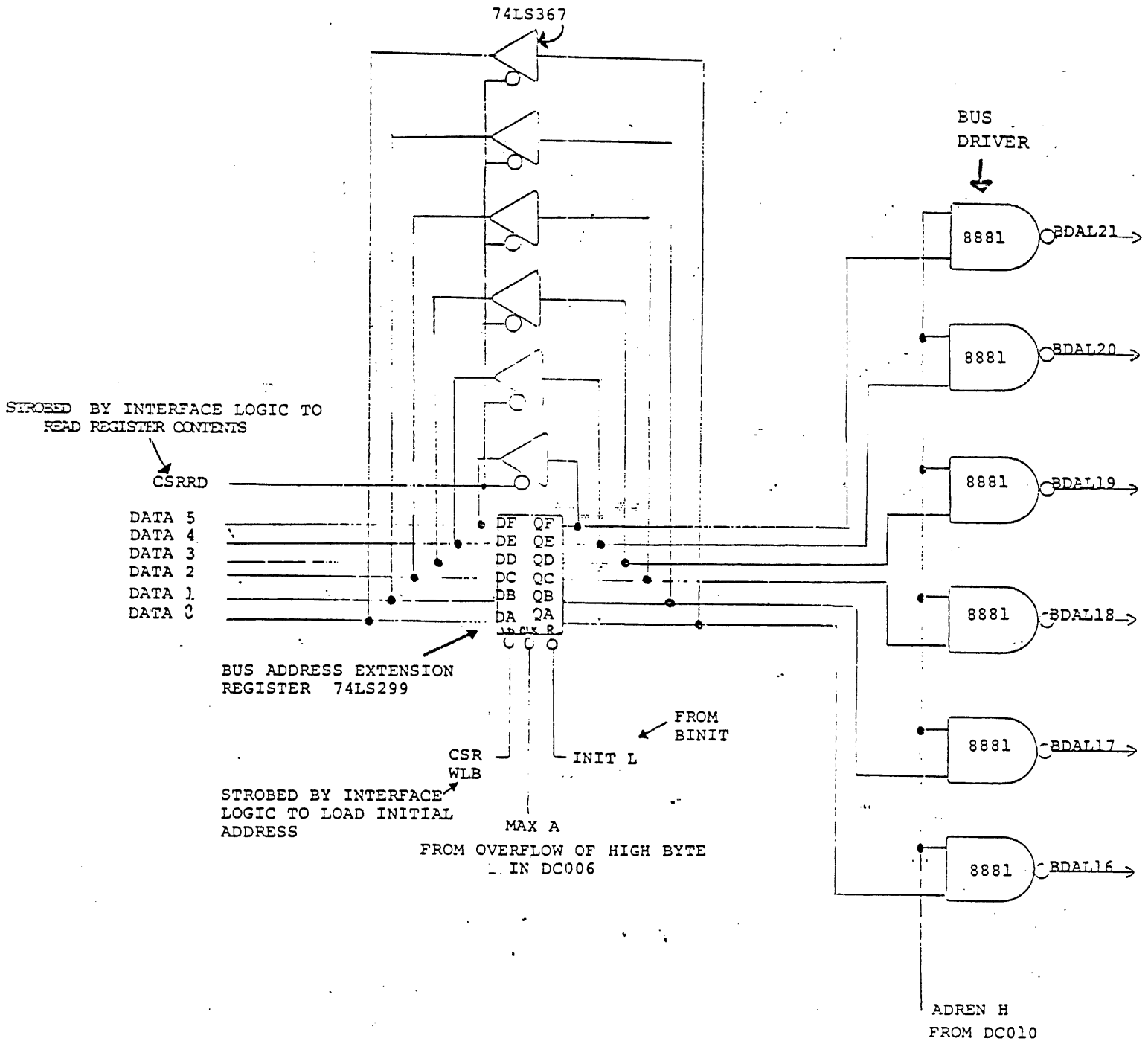
1. Provide a 6 bit address counter which will advance on overflow from the DC006. These six bits, along with the 16 bits from the DC006 will form the 22 bit address.
2. Provide for read/write of these 6 bits as bits 0-5 of a user-defined bus address extension register.
3. Provide for placing these 6 bits on address lines 16-21 during the address portion of the DMA transfer cycle.
4. Clear the additional bits upon initialization of the bus.

The following figure outlines the additional hardware. No extra chipkit hardware is needed. Signal mnemonics are consistent with those found in the Chipkit Users Manual.

This design does not consider an 18 bit or combination 18/22 bit addressing configuration. Information on designing an 18 bit interface can be found in uNote #69.

**digital**

**MICROCOMPUTER  
PRODUCTS  
GROUP**



<h1>note</h1> <p>TITLE <u>ADV11-A, AAV11-A, K WV11-A vs. ADV11-C, K WV11-C, AAV11-C Differences</u></p> <p>DISTRIBUTION <u>Unrestricted</u></p> <p>ORIGINATOR <u>Ernie Strange</u></p>	NUMBER 108
	DATE 01 / 05 / 82
	PRODUCT
	PAGE 1 OF 12

The following information compares the current analog products (ADV11-A, AAV11-A) with their replacement ADV11-C, AAV11-C. The real time clock K WV11-C and K WV11-A are functional equivalents. The terminology, as well as the specifications on both are identical.

Diagnostics used on the AAV11-A and K WV11 will run also on the AAV11-C and K WV11-C.

**digital**  
**MICROCOMPUTER**  
**PRODUCTS**  
**GROUP**

ADV11-A vs ADV11-C

NOTE: \*\* Represents significant differences  
\* Represents slight differences  
N/A Means information not published

	<u>ADV11-A</u>	<u>ADV11-C</u>
Resolution (A/D)	12 Bits	12 Bits
Linearity (A/D)	$\pm 1$ LSB (end point)	$\pm 1/2$ LSB
** Differential Linearity (A/D)	99% of State Widths $\pm 1/2$ LSB; No Skipped States; None > 2 LSB	Same
* Stability (A/D)	Gain - 6 PPM/ $^{\circ}$ C Linearity - 2 PPM/ $^{\circ}$ C Offset - 7.5 PPM/ $^{\circ}$ C	25 PPM/ $^{\circ}$ C Total
* Noise (A/D)	1/3 LSB RMS; 1 LSB Peak (module)  1/2 LSB RMS; 1.5 LSB Peak (in a system)	0.2mV. RMS
* Warm Up	5 mins. max.	N/A
* Thru-Put (A/D)	24,390 Channel Samples/Sec.	25,000 Channel Samples/Sec.
* Input Protection	Fuseable Resistors ( $\pm 85$ V.)	$\pm 30$ Volts
Coding	Offset Binary	Offset Binary or 2's Complement
** Input Range	$\pm 5.12$ V	$\pm 10$ V, 0 to +10V
** Software Programmable Ranges	None	X1, X2, X4, X8
* Number of Channels	16 Single-Ended (8 quasi-differential)	16 S.E 8 True Differential
Input Impedance	100M	100M



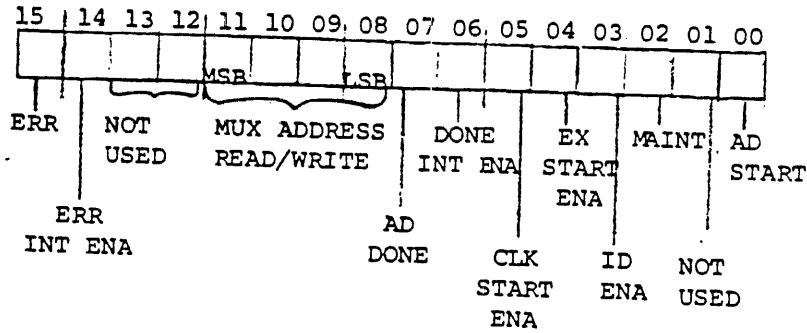
ADV11-A vs ADV11-C

Input Impedance (power off)	N/A	1 Kohm min.
Input Bias Current	50nA max.	15nA @ 25°C
* CMR	N/A	>80 db at 60 HZ
* S & H Aperture Uncertainty	N/A	<10nS
** Power Required	+5V/2A +12V/.45A	5V/1.5A
External Trigger	Hi-Lo Edge	Hi-Lo Edge
* Channel Select	Load Desired Channel Into Bits 8 - 11 of CSR (16 channels)	Bits 8 - 13 of CSR (up to 64 channels)
A/D Start	Bit 0, CSR	Bit 0, CSR
A/D Done	Bit 7, CSR	Bit 7, CSR
Error(s)	Bit 15, 14, CSR	Bit 15, 14, CSR
Interrupt Enable	Bit 6, CSR	Bit 6, CSR
Clock Enable	Bit 5, CSR	Bit 5, CSR
External Trigger Enable	Bit 4, CSR	Bit 4, CSR
** Bits 2, 3, CSR	I.D. Enable Maintenance	Programmable X1,X2,X4,X8 Gain Amplifier
CSR Base Address	VIA Dip Switch Range; 170000-177774	Wire Wrap 170000-177774
**	Data: Bits 0-11; I.D.= Bit 12 <u>Doubled Buffered</u>	Data: Bits 0-11 Bits 12 - 15 = MSB (signed extended)
Vector Address	Via Dip Switch 000-770	Wire Wrap 000-770
Operating Temperature	10° - 60°C	0° - 70°C
Input Connector (J1)	40 Pin Berg	26 Pin 3M



ADV11-A vs ADV11-C

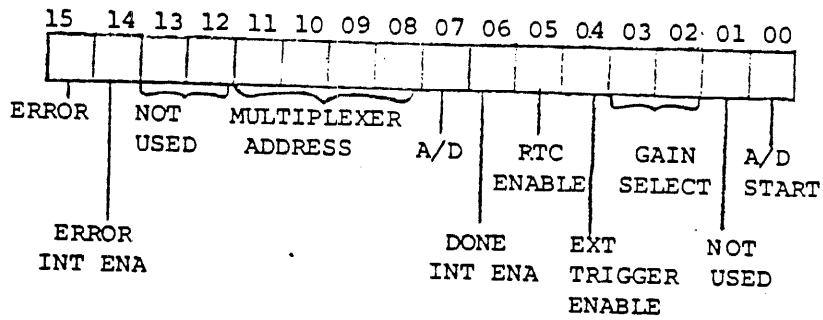
REGISTERS



ADV11-A CONTROL/STATUS REGISTER (CSR)

A/D CONTROL/STATUS REGISTER (READ/WRITE)

170400  
(BASE ADDRESS)



ADV11-C CONTROL/STATUS REGISTER (CSR)

Bits 2, 3 have a different function between ADV11-A and ADV11-C.

AAV11-A vs AXV11-C

The following specifications describe the two D/A channels on the AXV11-C. The DEC ADV11-A does not have D/A capability, therefore, the 2 D/A channels will be compared with the D/A specification of the AAV11-A four channel D/A.

	<u>AAV11-A</u>	<u>AXV11-C</u>
Resolution	12 Bits	12 Bits
*		
Output Voltage	$\pm 2.56, \pm 5.12, \pm 10.24$ 0 To +5.12, 0 To 10.24	$\pm 10V, 0$ To +10V
Differential Linearity	$\pm 1/2$ LSB Monotonic	$\pm 1/2$ LSB Monotonic
Linearity	$\pm 1/2$ LSB	$\pm 1/2$ LSB
**		
Slew Rate	5V/us	.33V/us
*		
D.C. Output Impedance	1	.05
**		
Rise Time	8us. to .1% of F.S.	35us. to .01% F.S. (for 10 volt change)

AAV11-A vs AAV11-C

Comparison of the AAV11-A and AAV11-C 4 channel D/A (the AAV11-A diagnostic also runs on the AAV11-C).

	<u>AAV11-A</u>	<u>AAV11-C</u>
Resolution	12 Bits	12 Bits
* Differential Linearity	$\pm 1/2$ LSB (monotonic)	$\pm 1/2$ LSB
Linearity	$\pm 1/2$ LSB	$\pm 1/2$ LSB
** Gain Drift	10 PPM/ $^{\circ}$ C	30 PPM/ $^{\circ}$ C Max
* Offset Drift	10 PPM/ $^{\circ}$ C	15 PPM/ $^{\circ}$ C Max
** Output Impedance	1	0.05
** Ranges	$\pm 2.65$ V, $\pm 5.12$ V, $\pm 10.24$ , 0 to 5.12V, 0 to 10.24V	$\pm 10$ V, 0 to $\pm 10$ V
* Output Current	5mA	10mA
* Slew Rate	5V/uSec.	20V/uSec.
* Settling Time	4us to 0.1%	For 20Volt Change, 6uSec Max. To within $\pm .1\%$ of Final Value
** Power	+5V/1.5A, 12V/.4A	+5V/2.0A
Bus Loading	1	1
** Size	Quad	Dual
Digital Output	4LSB's of DAC "3"	4LSB's of DAC "D"
Base Address Selection	Same Range for Both	

The real time clock, KWV11-A and the KWV11-C are functional equivalents. The terminology (as well as specifications) on both are identical. In fact, the KWV11-A diagnostic will run on the KWV11-C.

### ANALOG DEFINITION

RESOLUTION - The resolution of an A/D converter is defined as the smallest analog change that can be identified. Resolution is the analog value of the Least Significant Bit (LSB).

EX: A chemical processes can deliver 1 gal/min. It is desired to control flow to withing .2 of an oz.

$$\text{Resolution} = 1 \text{ LSB} = \frac{\text{Full Scale Range}}{2^{12} \text{ Code Combinations}}$$

$$1 \text{ Gal} = 128 \text{ oz.} \quad \frac{128 \text{ oz}}{4095} = .03125 \text{ oz.}$$

LINEARITY - Is defined as the maximum deviation from a straight line drawn between the end points of the converter transfer function.

EX: How close to a straight line are actual readings.

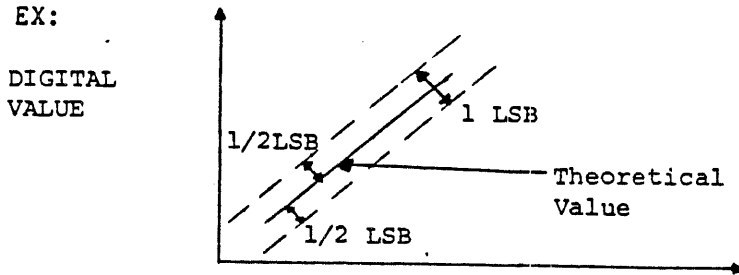


Given as a percentage of full scale or as a fraction of LSB. Usually at either extremes of range a non-linear (or deviation) value occurs.

## ANALOG DEFINITIONS

### DIFFERENTIAL LINEARITY -

The maximum deviation of an actual stated width from its theoretical value for any code over the full range of the converter. A differential linearity of  $1/2$  LSB means that the width of each code over the range of the converter is  $1 \text{ LSB} \pm 1/2 \text{ LSB}$ . Missing codes in an A/D converter occur when the output code skips a digit. This happens when the differential linearity is worse than  $\pm 1 \text{ LSB}$ .



STABILITY - How will the value change as temperature varies?

EX: Gain =  $6 \text{ PPM}/^{\circ}\text{C}$ .

The gain will not change more than 6 parts per million for a 1 degree centergrade.

NOISE - In an ideal system no noise is generated (zero noise). In the real world some noise will be introduced into the system through the cable, module (A/D converter), power supply.

THROUGHPUT - Is the number of samples per second that may be taken.

EX: System throughput =  $N(n) \times (\text{B.W})$  samples/sec

Channel 1 = 150 HZ

Channel 2 = 800 HZ

Channel 3 = 400 HZ

System throughput =  $N(n) \times (\text{Highest Band Width})$  samples/sec

$N = 10$  (Sampling Factor)

Normal from 5 - 10 samples per cycle are needed to give a reasonable reproduction of sampled signal

$n = 3$  (Number of Channels)

BW = 800

Throughput =  $(10) (3) (800)$

= 24K Samples/Second

## ANALOG DEFINITIONS

### CODING

BINARY	<u>Full Scale Input Voltage</u>	<u>Output Code (Octal)</u>
	+9.9976V	007777
	0.00000	000000

The binary numbers from  $0_{(8)}$  to  $007777_{(8)}$  represent the input voltages 0V to 9.9976V.

BINARY OFFSET	<u>Full Scale Input Voltage</u>	<u>Output Code (Octal)</u>
	+9.9951V	007777
	0.0000V	004000
	-10.0000V	000000

Using the binary offset technique, a zero voltage point is established at  $004000_{(8)}$ . This technique is usually used when both positive and negative voltages are present.

TWO'S COMPLEMENT	<u>Full Scale Input Voltage</u>	<u>Output Code (Octal)</u>
	+9.9951V	003777
	0.0000V	000000
	-10.0000V	174000

Using Two's Complement, 0V in is represented by  $000000_{(8)}$  in the output code. -10V in is represented by  $174000_{(8)}$ . The output code range is still  $007777_{(8)}$  it's just formatted differently.



## ANALOG DEFINITIONS

INPUT RANGE - The range of voltages that can be applied to the input of an A/D converter. If this range is exceeded damage may result to A/D converter.

### SOFTWARE PROGRAMMABLE RANGES

- Allows the resolution to be increased by changing the full scale voltage range.

EX: If the gain is set at one and the input voltage is from 0 - 10 volts;

$$\text{Resolution} = \frac{\text{Full Scale Voltage Range}}{2^{12}}$$

$$\text{Resolution} = \frac{10\text{V}}{4095} = 2.442 \text{ mv}$$

Changing the programmable gain to 8 allows the input voltage range of 0 - 1.25V.

$$\text{Resolution} = \frac{1.25\text{V}}{4095} = 305.2 \text{ mv}$$

### NUMBER OF CHANNELS

- Channels are a communication path thru which analog voltages are translated to Digital information. The number of channels specify how many analog voltages may be monitored.

### INPUT IMPEDANCE

- The amount of internal impedance of the input channel. The higher the impedance the less loading on the external circuit occurs. Ideally the input impedance would be infinite so no current would be drawn from the external source.

## ANALOG DEFINITIONS

- INPUT BIAS  
CURRENT - The amount of current that flows into the selected A/D channel from the source.
- CMR - Common Mode Rejection is the ability of a differential amplifier to reject noise common to both inputs.
- S & H  
APERTURE  
UNCERTAINTY - Sample and Hold Aperture Uncertainty is the change in aperture delay times between specific sample and hold commands.
- CHANNEL  
SELECT - The control status register has bits (8 - 11) dedicated to selecting one of 16 possible channels.
- SLEW RATE - The capability of the output of an analog circuit to change its voltage in a given period of time.
- EX: If the slew rate is 5V/uSec, the analog circuit output will change five volts in one uSecond.
- GAIN DRIFT - Is the amount of change in gain due to the temperature co-efficient of the components.
- OFFSET DRIFT - Is a function of the temperature co-efficients of the components.

<b>note</b>		NUMBER 109
TITLE	<u>Using the FALCON SBC-11/21 in a Standalone Environment</u>	DATE 04 / 13 / 82
DISTRIBUTION	<u>Unrestricted</u>	PRODUCT FALCON
ORIGINATOR	<u>Charlie Giorgetti</u>	PAGE 1 OF 2

The FALCON SBC-11/21 can be used in standalone applications. The user must interface to the module with power via a connector block and then mount the FALCON SBC-11/21. Environmental requirements should be adhered to when the board is mounted (see the Microcomputer Handbook Series for environmental specifications).

The mounting holes for the FALCON SBC-11/21 are shown in Figure 1. The FALCON SBC-11/21 should be mounted using the holes provided. When the actual mounting is done, insulated washers should be used to isolate the screws from the module surface. Insulated standoffs should be used to raise the FALCON SBC-11/21 off the surface to which it is being secured. The standoffs should have the following characteristics: 0.15 inch diameter, .1 inch hole size, and a height of at least 0.75 inches. The connector block that is recommended for use with the FALCON SBC-11/21 is DEC part number H8030. This is a 72 pin block that fits over the module edge connector and allows a user to access a finger via wirewrap pins that are on the connector block.

Since the FALCON SBC-11/21 supports on-board wake-up circuitry, the +5 and +12 volts can be obtained from any supply which is capable of meeting the module's power requirements. A power supply, such as the DEC H780, can be used in those applications that require power fail detection. The needed voltages can be applied to the connector block on the following wirewrap pins:

- GND            AM1, AC2, AT1, AJ1, BC2, BT1, BM1, or BJ1
- +5             AA2, BA2, or BV1
- +12            BD2

Refer to the Microcomputer Handbook Series for the locations of each wirewrap pin.

It is of interest that in FALCON SBC-11/21 applications that are not using the serial line units only +5 volts is required.

For applications that are using battery back-up for the on-board static RAM, in the event of a power failure, +5BB is applied to pin AV1 on the connector block.



*μnote*

NUMBER 109  
PAGE 2 OF 2

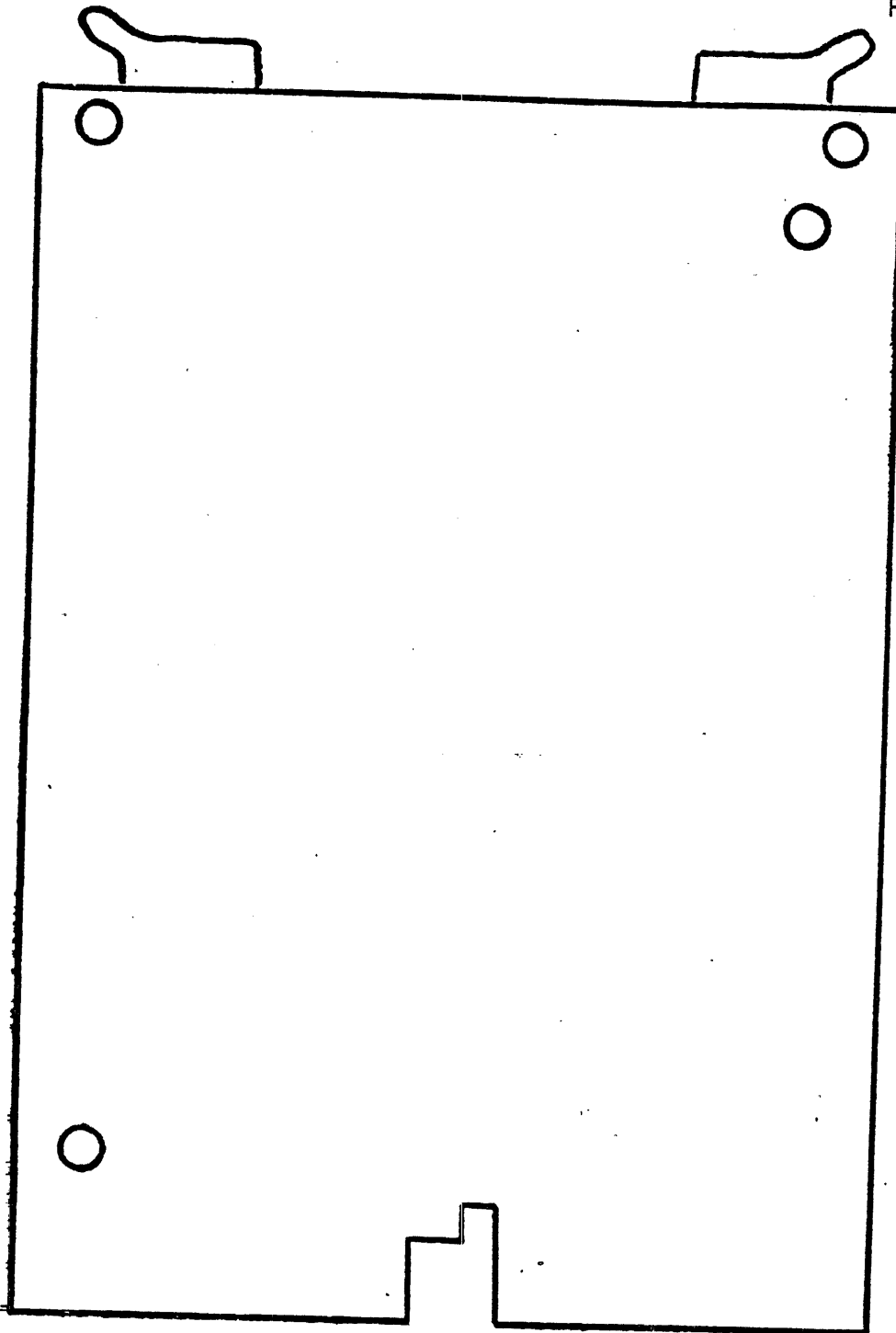


Figure 1: COMPONENT SIDE VIEW OF FALCON SBC-11/21

**digital**  
MICROCOMPUTER  
PRODUCTS  
GROUP

<b><i>μ</i>note</b>		NUMBER 110
TITLE <u>MUL, DIV, and ASH Instruction for the FALCON</u> <u>SBC-11/21</u>		DATE 04 / 13 / 82
DISTRIBUTION <u>Unrestricted</u>		PRODUCT FALCON
ORIGINATOR <u>Charlie Giorgetti</u>		PAGE 1 OF 4

The FALCON SBC-11/21 supports the standard PDP-11 instruction set. There is no hardware support for the EIS, FIS or FPP instruction sets. For FALCON SBC-11/21 applications that need the ability to perform the EIS instructions MUL, DIV, and ASH, equivalent software routines can be substituted. These callable routines do not do any form of error checking. A user should be aware that extensive use of these software routines for hardware instructions will have impact on system performance.

These routines can be incorporated into an application and called as a subroutine. The calling sequence for the subroutines can be set-up in a macro. The following is a list of each of the subroutines and the macros that are used to set-up and call the software MUL, DIV, and ASH routines.

The following macro and subroutine can be used to perform the MUL instruction in software:

```

.MACRO SMUL A,B,HI,LO

MOV A,-(SP) ; Push a multiplier onto the stack
MOV B,-(SP) ; Push the other multiplier as well
JSR PC,SMUL ; Call the MUL subroutine
MOV (SP)+,HI ; Get the most significant part of the result
MOV (SP)+,LO ; Get the least significant part of the result

.ENDM

SMUL:: MOV R0,-(SP) ; Save some work registers
MOV R1,-(SP)
MOV 10(SP),R1 ; Obtain the value of A from the stack
MOV #21,-(SP) ; Initialize the shift counter
CLR R0 ; Initialize the high 16-bit accumulator
10s: ROR R0 ; Perform multiplication
ROR R1
BCC 20s
ADD 10(SP),R0
CLC
20s: DEC (SP) ; Bump the shift counter
BNE 10s ; Not done ?
TSI (SP)+ ; Remove the counter from the stack
MOV R1,10(SP) ; Save the low 16-bit value on the stack
MOV R0,6(SP) ; Save the high 16-bit value on the stack
MOV (SP)+,R1 ; Restore the work registers
MOV (SP)+,R0
RIS PC ; Return

```

The following macro and subroutine can be used to perform the DIV Instruction in software:

```

.MACRO SDIV DIVSOR,DIVHI,DIVLO,REM,QUO

MOV    DIVSOR,-(SP)    ; Push the divisor onto the stack
MOV    DIVHI,-(SP)    ; Push the upper 16-bits of the dividend
MOV    DIVLO,-(SP)    ; Push the lower 16-bits of the dividend
JSR    PC,SDIV        ; Call the DIV subroutine
MOV    (SP)+,REM      ; Get the remainder
MOV    (SP)+,QUO      ; Get the quotient

.ENDM

SDIV:: MOV    R5,-(SP)    ; Get some work registers
      MOV    R4,-(SP)
      MOV    R3,-(SP)
      MOV    R0,-(SP)
      MOV    14.(SP),R3    ; Get the divisor from the stack
      MOV    12.(SP),R4    ; Get the high 16-bits of the dividend
      MOV    10.(SP),R5    ; as well as low 16-bits
      CLR    R0            ; Clear an accumulator
      MOV    #32.,-(SP)    ; Shift counter
1s:   ASL    R5            ; Perform the division
      ROL    R4
      ROL    R0
      CMP    R0,R3
      BLO   2s
      SUB    R3,R0
      INC   R5
2s:   DEC   (SP)
      BNE   1s            ; Not done ?
      IST   (SP)+        ; Remove the counter from the stack
      MOV   R0,12.(SP)    ; Store the remainder on the stack
      MOV   R5,14.(SP)    ; Store the quotient as well
      MOV   (SP)+,R0      ; Restore the work registers
      MOV   (SP)+,R3
      MOV   (SP)+,R4
      MOV   (SP)+,R5
      MOV   (SP)+,(SP)    ; Update the return PC
      RTS   PC            ; Return

```

The following macro and subroutine can be used to perform the ASH instruction in software:

```

.MACRO SASH COUNT,VAL

MOV COUNT,-(SP) ; Push the shift count
MOV VAL,-(SP) ; Push what is to be shifted
JSR PC,$ASH ; Call the ASH subroutine
MOV (SP)+,VAL ; Get the results of the shift

.ENDM


SASH:: MOV R0,-(SP) ; Get a couple of work registers
MOV R1,-(SP)
MOV 6(SP),R0 ; R0 = value to be shifted
MOV 8.(SP),R1 ; R1 = direction and shift count
BIC #^C<77>,R1
BEQ 20$ ; Get out if no shifting
CMP R1,#31. ; what direction is the shift
BGT 10$ ; Go to the corection direction shift
5$: ASL R0
DEC R1
BNE 5$
BR 20$
10$: NEG R1
BIC #^C<77>,R1
11$: ASL R0
DEC R1
BNE 11$
20$: MOV R0,8.(SP) ; Store the shifted result on the stack
MOV (SP)+,R1 ; Restore the work registers
MOV (SP)+,R0
MOV (SP)+,(SP) ; Update the return PC
RTS PC ; Return

```

**digital**

**MICROCOMPUTER  
PRODUCTS  
GROUP**



 TITLE <u>DIFFERENCES BETWEEN MSV11-L and MSV11-P MEMORIES</u> DISTRIBUTION <u>Unrestricted</u> ORIGINATOR <u>Mike Collins</u>	NUMBER 111
	DATE 05 / 04 / 82
	PRODUCT
	PAGE 1 OF 2

There are now two series of memory boards for the LSI-11 bus which have full parity functionality on-board and also utilize the 22 bit addressing capability. They are the MSV11-L and the MSV11-P.

This uNOTE will discuss differences existing between the MSV11-L and P memories which should be considered when choosing and configuring these boards for a system.

DIFFERENCE: MSV11-L is a dual height module.  
MSV11-P is a quad height module.

The MSV11-P memory is physically twice the size of the L memory and requires a quad size backplane (e.g. the H9275-A, which is a 9 x 4 backplane).

DIFFERENCE: MSV11-L has 8 CSRs (Control Status Register).  
MSV11-P has 16 CSRs.

The CSR allows the user to enable parity error reporting and will also report the error to within a 2 Kbyte segment of memory.

With MSV11-L memories, full parity functionality can be utilized on up to a maximum of 2 Megabytes. The MSV11-P memories have 8 additional CSRs which allows the complete 4 Megabytes of address space to use the full parity feature.

DIFFERENCE: MSV11-L memories are configurable on 8 Kbyte boundaries.  
MSV11-P memories are configurable on 16 Kbyte boundaries.

This information must be kept in mind when configuring the boards in a system. Previous to the MSV11-P, all other memory options were configureable on 8Kb boundaries. This includes the MSV11-C, D and E memories.

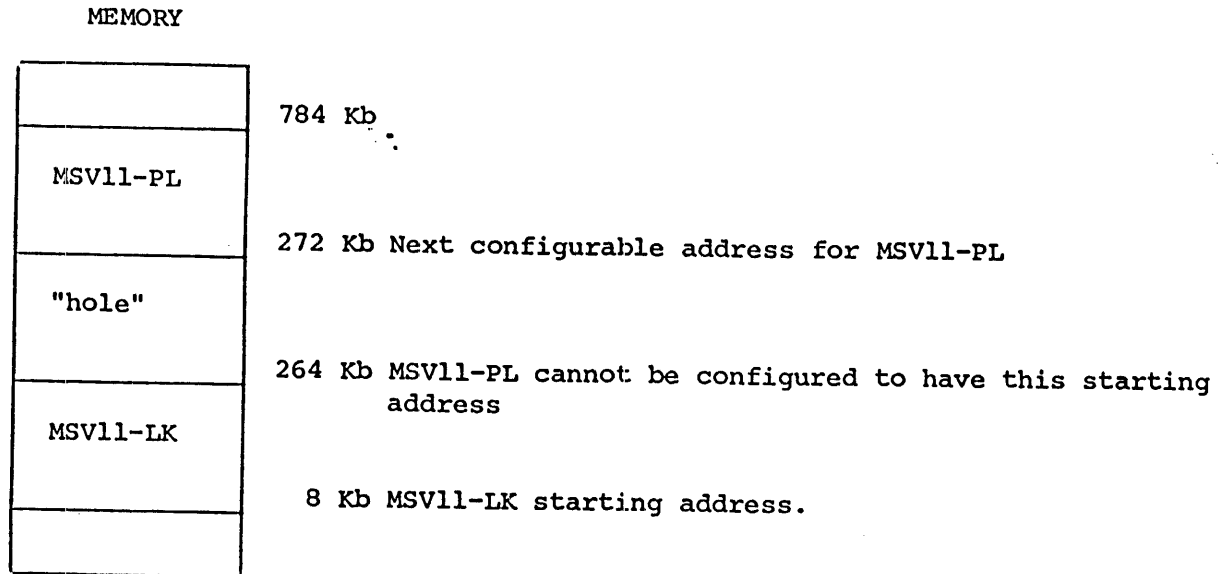
No problems would arise if only L and P memories are used in a system and they are configured contiguously from a starting address of zero. However, a problem could occur if all of the following conditions exist:

Both L and P memories are to be used in a system. The memories are to be contiguous and an L memory is configured first with a starting address on an ODD 8 Kb boundary.

**digital**  
**MICROCOMPUTER**  
**PRODUCTS**  
**GROUP**

For example: A system uses 1 MSV11-LK memory (256 Kb) and  
1 MSV11-PL memory (512 Kb).

If for some reason the MSV11-LK memory is configured for a starting address of 8 Kb, its last address is 264 Kb (256 Kb + 8 Kb). But the MSV11-PL cannot have a starting address of 264 Kb because it is only configurable on 16 Kb boundaries. If the P memory is configured for the next possible starting address, a "hole" would exist in the memory space, and the two memories would not be contiguous.



**DIFFERENCE:** It is possible to disable half of the MSV11-L RAM array. There are no provisions for disabling RAM on the P memory module.

If bad memory exists in one of the two 128 Kb sections of the MSV11-LK, the bad section can be disabled. The board will then look like a half populated module and the address will be from 0 Kbyte to 128 Kbytes. As stated above, there are no jumpers on the MSV11-P memory which allow selective disabling of the RAM array.

<b>note</b>		NUMBER 112
TITLE	18-BIT RXV21 in 22-BIT LSI-11 RSX-11M V4.0	DATE 2 / 17 / 83
DISTRIBUTION	Un-restricted	PRODUCT
ORIGINATOR	Bernie Alimonti	PAGE 1 OF 4

Incorporating an 18-BIT RXV21 Controller  
Into a 22-Bit LSI-11 RSX-11M V4.0 System

The LSI-11 Bus specification has always provided for 22-bit addressing. However, only since spring of 1982, has DIGITAL acknowledged and supported 22-bit addressing on the LSI-11 Bus. The RL01/RL02 mass storage controller for the LSI-11 Bus was redesigned in 1982 as the RLV12, and includes support for 16-, 18- as well as 22-Bit addressing. The RX02 mass storage controller for the LSI-11 Bus, the RXV21, however, does not support 22-bit addressing.

RSX-11M V4.0 supports 22-bit addressing on LSI-11 systems, and supports the RLV12 in a 22-bit LSI-11. However, it does not support the RXV21 in a 22-bit LSI-11. The reason for this is that without 22-bit addressing the RXV21 cannot perform DMA into the total range of addresses. Below is a discussion of how you can, in fact, incorporate an RXV21 into your LSI-11 running a 22-bit RSX-11M V4.0 system.

The RSX-11M RX02 driver, DYDRV.MAC, assumes that if 22-bit addressing is performed, it is performed only on a UNIBUS system. With this assumption, 22-bit DMA is performed using UNIBUS mapping registers (UMR). If the code for the UMRs is used in a 22-bit LSI-11 system, RSX-11M will not operate properly. Therefore, the approach taken here to incorporate an RXV21 into a LSI-11 22-bit running RSX-11M is to modify DYDRV.MAC not to use UMRs. Together with the driver change, tasks that do I/O to the RX02 are restricted to reside only in the first 256KB of memory. The modified RX02 driver should not be used in 18-bit LSI-11 or UNIBUS system, nor a 22-bit UNIBUS system.

**digital**  
**MICROCOMPUTER**  
**PRODUCTS**  
**GROUP**

The correction file below for DYDRV.MAC is for the distribution (V4.0 master) version of the file. It does not include any other driver corrections. You can easily integrate this correction with any another correction(s) for DYDRV.MAC.

Although this information is provided by DIGITAL, it is not authored by the RSX-11M development group, and therefore is not officially supported by DIGITAL.

### Steps to Include an 18-bit RXV21 Controller Into a 22-bit LSI-11 RSX-11M V4.0 system

1. Create the following SLP correction file for DYDRV.MAC and modify the driver source using the procedure outlined in the first paragraph (and example) of Section 5.2.1 (page 46) of the RSX-11M V4.0 Release Notes.

```
DYDRV.MAC;2/AU:72./-BF=[11,10]DYDRV.MAC;1/CS:62064
\
-2,.
      .IDENT    /02.16X/
-10,.
; VERSION 02.16X
-76
;
;      B. ALIMONTI      14-SEP-82
;      BA160--MODIFY FOR 18 BIT DY CONTROLLER ON 22 BIT SYSTEM
;
-308,315,;/;BA160/
      .IF DF M$$EIS

      MOV      U.BUF(R5),R0      ; MOV ADR EXTENSION BITS TO R0
      ASH      #4,R0             ; MOV ADR BITS 16 AND 17 INTO POS
      BIC      #^C<30000>,R0    ; ISOLATE ADR BITS 16 AND 17
      MOV      R0,U.BUF(R5)      ; REPLACE BACK IN U.BUF

      .IFF

      .REPT 4
      ASL      U.BUF(R5)        ; MOV ADR BITS 16 AND 17 INTO POS
      .ENDR
```

```

BIC #^C<30000>,U.BUF(R5) ; ISOLATE ADR BITS 16 AND 17

.ENDC ;M$EIS
-343,353,/,;BA160/
;
/

```

2. If you have already performed a SYSGEN and your RX02 driver is loadable, follow the procedure outlined in Section 5.1.2 of the Release Notes for loadable drivers (page 47) to reassemble and rebuild the driver. Do not yet VMR the system as indicated in the Release Notes (but instead go to step 4 below).
3. Perform a SYSGEN if: 1) you have not yet done so, or 2) you have performed a SYSGEN but the RX02 driver is resident. Use the modified driver source to do the SYSGEN.
4. Create a new system image as follows:

- a) Set the UIC to [1,54] with the command  
SET /UIC=[1,54]
- b) Edit SYSVMR.COM to establish a partition, 18BIT, in the first 256KB of memory for those tasks that perform I/O to the RX02. You do this by first locating the line

```
SET /MAIN=FCPPAR*:fcplen:SYS
```

and the adding the line

```
SET /MAIN=18BIT*:18bitlen:SYS
```

following it , where fcplen is the length of partition FCPAR, and 18bitlen is the length of partition 18BIT (as long as partition 18BIT is contained totally within the first 256KB).

c) Create a new system image file with the command

```
PIP RSX11M.SYS/NV/CO/BL:498.=RSX11M.TSK
```

d) VMR the system as follows

```
VMR @SYSVMR.CMD
```

4. Install those tasks that perform I/O to the RX02 into partition 18BIT. This is done either with VMR or MCR. For example

```
INS PIP/PAR=18BIT
```

<b>ynote</b>		NUMBER
		113
TITLE	<u>BLOCK MODE DMA</u>	DATE
		6 / 1 / 83
DISTRIBUTION	<u>Unrestricted</u>	PRODUCT
ORIGINATOR	<u>Mike Collins, Scott Tincher</u>	PAGE 1 OF 15

## BLOCK MODE DMA

### What is Block Mode DMA?

Block mode DMA is a method of data transfer which increases throughput due to the reduced handshaking necessary over the Q-bus. In order to implement Block mode DMA both the master and slave devices must understand the block mode protocol. If either device does not have Block mode capability the transfers proceed via standard DATI or DATO cycles.

### Conventional Direct Memory Access on the Q-bus.

Under conventional DMA operations, after a DMA device has become bus master, it begins the data transfers. This is accomplished by gating an address onto the bus followed by the data being transferred to or from the memory device. If more than one transfer is performed by the temporary bus master, the address portion of the cycle must be repeated for each data transfer.

### Block Mode Direct Memory Access on the Q-bus.

Under block mode DMA operations an address cycle is followed by multiple word transfers to sequential addresses. Therefore data throughput is increased due to the elimination of the address portion of each transfer after the initial transfer.

There are two types of block mode transfer, DATBI (input) and DATBO (output). An overview of what occurs during each type of block mode transfer is outlined in figures 1 (DATBI, block mode input) and 2 (DATBO, block mode output).

A detailed description of each type of transfer accompanies figures 1 and 2 as well as detailed timing diagrams.

In the following discussion the signal prefix T(Transmit) indicates a bus driver input and the signal prefix R(Receive) indicates a bus receiver output.

**digital**

## DATBI Bus cycles

Before a block mode transfer can occur the DMA bus master device must request control of the bus. This occurs under conventional Q-bus protocol.

- o REQUEST BUS  
The bus master device requests control of the bus by asserting TDMR.
- o GRANT BUS CONTROL  
The bus arbitration logic in the CPU asserts the DMA grant signal TDMGO 0 nsec minimum after TDMR is received and 0 nsec minimum after RSACK negates (if a DMA device was previous bus master).
- o ACKNOWLEDGE BUS MASTERSHIP  
The DMA bus master device asserts TSACK 0 nsec minimum after receiving RDMGI, 0 nsec minimum after the negation of RSYNC and 0 nsec minimum after the negation of RRPLY. The DMA bus master device negates TDMR 0 nsec minimum after the assertion of TSACK.
- o TERMINATE GRANT SEQUENCE  
The bus arbitration logic in the CPU negates TDMGO 0 nsec minimum after receiving RSACK. The bus arbitration logic will also negate TDMGO if RDMR negates or if RSACK fails to assert within 10 usec ('no SACK timeout').
- o EXECUTE A BLOCK MODE DATBI TRANSFER
  - o ADDRESS DEVICE MEMORY
    - a) The address is asserted by the bus master on TADDR<21:00> along with the negation of TWTBT.
    - b) The bus master asserts TSYNC 150 nsec minimum after gating the address onto the bus.
  - o DECODE ADDRESS  
The appropriate memory device recognizes that it must respond to the address on the bus.
  - o REQUEST DATA
    - a) The address is removed by the bus master from TADDR<21:00> 100 nsec minimum after the assertion of TSYNC.
    - b) The bus master asserts the first TDIN 100 nsec minimum after asserting TSYNC.



- c) The bus master asserts TBS7 50 nsec maximum after asserting TDIN for the first time. TBS7 remains asserted until 50 nsec maximum after the assertion of TDIN for the last time. In each case, TBS7 can be asserted or negated as soon as the conditions for asserting TDIN are met.

The assertion of TBS7 indicates the bus master is requesting another read cycle after the current read cycle.

- o SEND DATA

- a) The bus slave asserts TRPLY 0 nsec minimum (8000 nsec maximum to avoid a bus timeout) after receiving RDIN.
- b) The bus slave asserts TREF concurrent with TRPLY if, and only if, it is a block mode device which can support another RDIN after the current RDIN.

NOTE

Block mode transfers must not cross 16  
word boundaries

- c) The bus slave gates TDATA<15:00> onto the bus 0 nsec minimum after receiving RDIN and 125 nsec maximum after the assertion of TRPLY.

- o TERMINATE INPUT TRANSFER

- a) The bus master receives stable RDATA<15:00> from 200 nsec maximum after receiving RRPLY until 20 nsec minimum after the negation of RDIN. (The 20 nsec minimum represents total minimum receiver delays for RDIN at the slave and RDATA<15:00> at the master.)
- b) The bus master negates TDIN 200 nsec minimum after receiving RRPLY.

- o OPERATION COMPLETED

- a) The bus slave negates TRPLY 0 nsec minimum after receiving the negation of RDIN.
- b) If RBS7 and TREF are both asserted when TRPLY negates, the bus slave prepares for another DIN cycle. RBS7 is stable from 125 nsec after RDIN is received until 150 nsec after TRPLY negates.

- c) If TBS7 and RREF were both asserted when TDIN negated, the bus master asserts TDIN 150 nsec minimum after receiving the negation of RRPLY and continues with timing relationship 'SEND DATA' above. RREF is stable from 75 nsec after RRPLY asserts until 20 nsec minimum after TDIN negates. (The 0 nsec minimum represents total minimum receiver delays for RDIN at the slave and RREF at the master.)

### Note

The bus master must limit itself to not more than eight transfers unless it monitors RDMR. If it monitors RDMR, it may perform up to 16 transfers as long as RDMR is not asserted at the end of the seventh transfer.

- o TERMINATE BUS CYCLE

- a) If RBS7 and TREF were not both asserted when TRPLY negated, the bus slave removes TDATA<15:00> from the bus 0 nsec minimum and 100 nsec maximum after negating TRPLY.
- b) If TBS7 and RREF were not both asserted when TDIN negated the bus master negates TSYNC 250 nsec minimum after receiving the last assertion of RRPLY and 0 nsec minimum after the negation of that RRPLY.

- o RELEASE THE BUS

- a) The DMA bus master negates TSACK 0 nsec after negation of the last RRPLY.
- b) The DMA bus master negates TSYNC 300 nsec maximum after it negates TSACK.
- c) The DMA bus master must remove RDATA<15:00>, TBS7, and TWTBT from the bus 100 nsec maximum after clearing TSYNC.

- o RESUME PROCESSOR OPERATION

The bus arbitration logic in the CPU enables processor-generated TSYNC or will issue another bus grant (TDMGO) if RDMR is asserted.

D A T B I C Y C L E

PROCESSOR

I/O DEVICE

MEMORY

Request Bus  
 . Assert TDMR

Grant Bus Control

. Near end of the current bus cycle (RRPLY is negated) assert TDMGO and inhibit new processor generated TSYNC for the duration of the DMA operation

Acknowledge Bus  
Mastership

. Receive RDMGO  
 . Wait for negation of RSYNC and RRPLY  
 . Assert TSACK  
 . Negate TDMR

Terminate Grant Sequence

. Negate RDMGO and wait for DMA operation to be completed.

Execute a Block Mode DMA (DATBI) Data Transfer

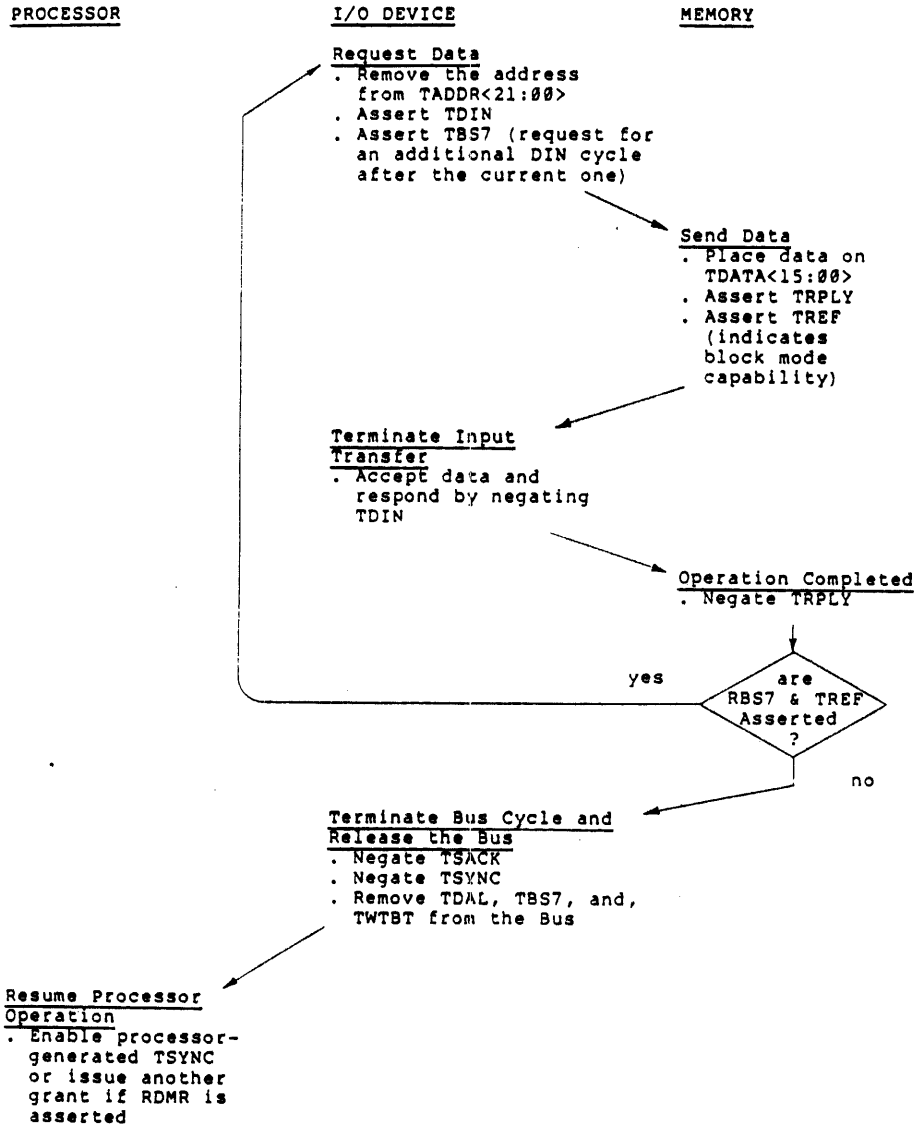
.  
 .  
Address Device  
Memory

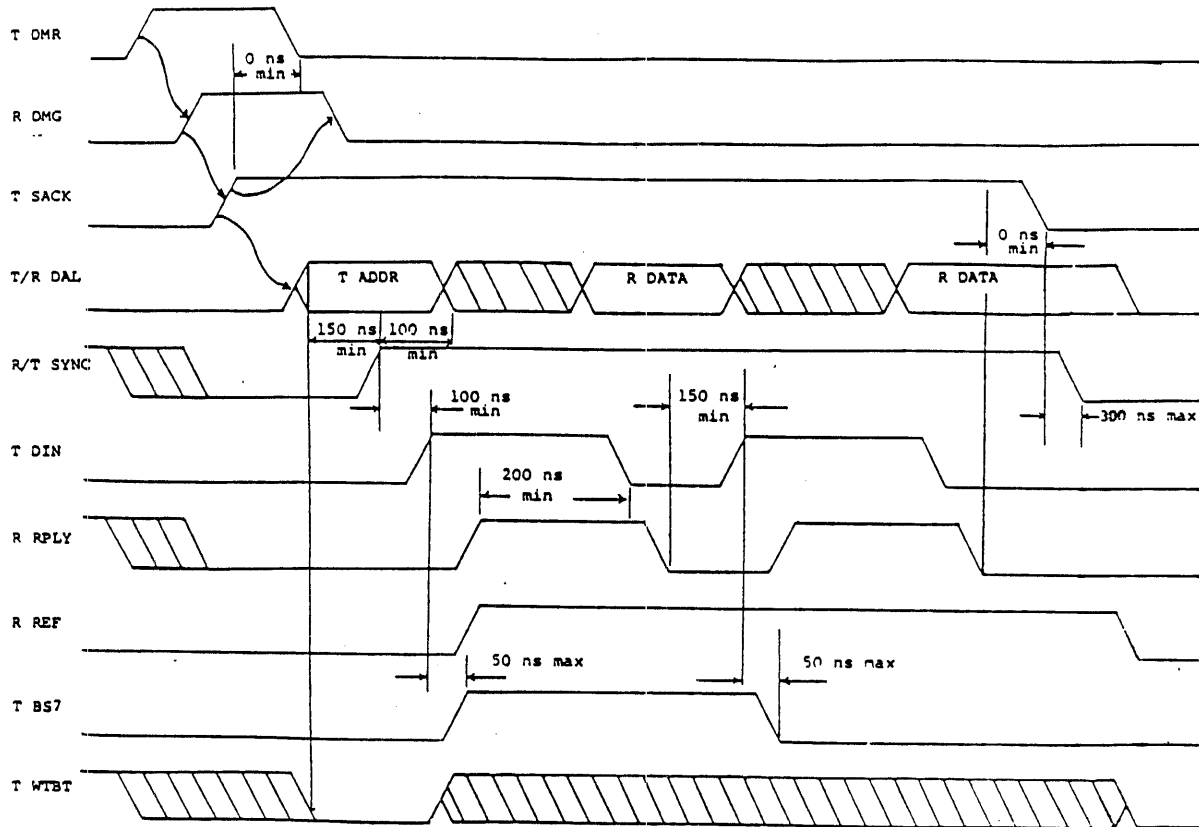
. Assert TADDR<21:00> with address  
 . Assert TSYNC  
 . Negate TWTBT

Decode Address

. Store "Device Selected" operation

DATBI CYCLE CONT'D

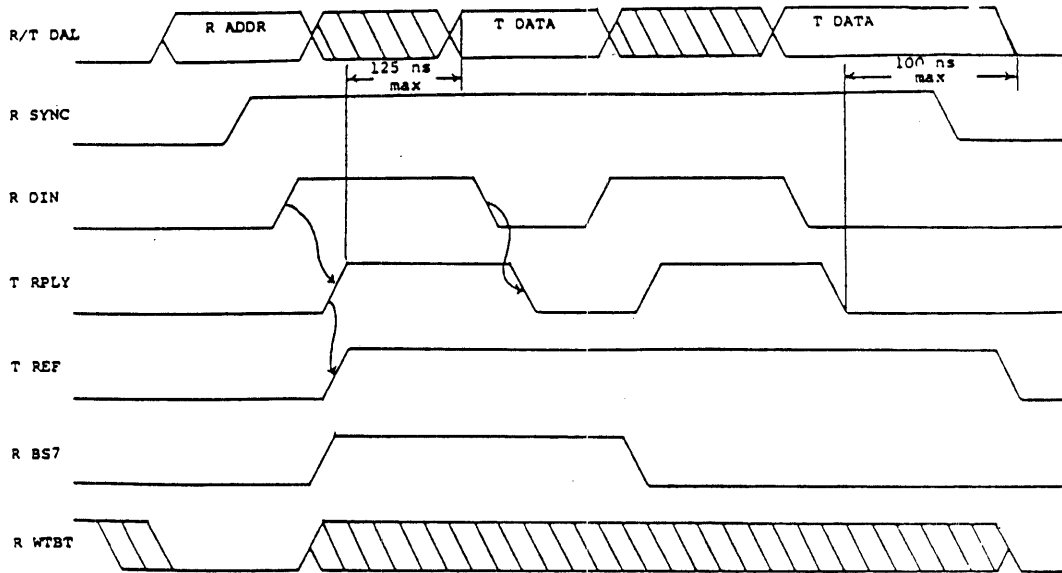




Timing at master device.  
T = Bus driver input  
R = bus receiver output

DATBI

digital



Timing at slave device.  
T = Bus driver input  
R = Bus receiver output

DATBI

**digital**

## DATBO Bus cycles

Before a block mode transfer can occur the DMA bus master device must request control of the bus. This occurs under conventional Q-bus protocol.

### o REQUEST BUS

The bus master device requests control of the bus by asserting TDMR.

### o GRANT BUS CONTROL

The bus arbitration logic in the CPU asserts the DMA grant signal TDMGO 0 nsec minimum after RDMR is received and 0 nsec minimum after TSACK negates (if a DMA device was previous bus master).

### o ACKNOWLEDGE BUS MASTERSHIP

The DMA bus master device asserts TSACK 0 nsec minimum after receiving RDMGI, 0 nsec minimum after the negation of RSYNC and 0 nsec minimum after the negation of RRPLY. The DMA bus master device negates TDMR 0 nsec minimum after the assertion of TSACK.

### o TERMINATE GRANT SEQUENCE

The bus arbitration logic in the CPU negates TDMGO 0 nsec minimum after receiving RSACK. The bus arbitration logic will also negate TDMGO if RDMR negates or if RSACK fails to assert within 10 usec ('no SACK timeout').

### o EXECUTE A BLOCK MODE DATBO TRANSFER

#### o ADDRESS DEVICE MEMORY

a) The address is asserted by the bus master on TADDR<21:00> along with the assertion of TWTBT.

b) The bus master asserts TSYNC 150 nsec minimum after gating the address onto the bus.

#### o DECODE ADDRESS

The appropriate memory device recognizes that it must respond to the address on the bus.

#### o SEND DATA

a) The bus master gates TDATA<15:00> along with TWTBT 100 nsec minimum after the assertion of TSYNC. TWTBT is negated.

- b) The bus master asserts the first TDOUT 100 nsec minimum after gating TDATA<15:00>.

NOTE

During DATBO cycles TBS7 is undefined

o RECEIVE DATA

- a) The bus slave receives stable data on RDATA<15:00> from 25 nsec minimum before receiving RDOUT until 25 nsec minimum after receiving the negation of RDOUT.
- b) The bus slave asserts TRPLY 0 nsec minimum after receiving RDOUT.
- c) The bus slave asserts TREF concurrent with TRPLY if, and only if, it is a block mode device which can support another RDOUT after the current RDOUT.

NOTE

Blockmode transfers must not cross 16 word boundaries

o TERMINATE OUTPUT TRANSFER

The bus master negates TDOUT 150 nsec minimum after receiving RRPLY.

o OPERATION COMPLETED

- a) The bus slave negates TRPLY 0 nsec minimum after receiving the negation of RDOUT.
- b) If RREF was asserted when TDOUT negated and if the bus master wants to transfer another word, the bus master gates the new data on TDATA<15:00> 100 nsec minimum after negating TDOUT. RREF is stable from 75 nsec maximum after RRPLY asserts until 20 nsec minimum after RDOUT negates. (The 20 nsec minimum represents minimum receiver delays for RDOUT at the slave and RREF at the master).



- c) The bus master asserts TDOUT 100 nsec minimum after gating new data on TDATA<15:00> and 150 nsec minimum after receiving the negation of RRPLY. The cycle continues with the timing relationship in 'RECEIVE DATA' above.

## Note

The bus master must limit itself to not more than eight transfers unless it monitors RDMR. If it monitors RDMR, it may perform up to 16 transfers as long as RDMR is not asserted at the end of the seventh transfer.

- o TERMINATE BUS CYCLE

- a) If RREF was not asserted when RRPLY negated or if the bus master has no additional data to transfer, the bus master removes data on TDATA<15:00> from the bus 100 nsec minimum after negating TDOUT.
- b) If RREF was not asserted when TDOUT negated the bus master negates TSYNC 275 nsec minimum after receiving the last RRPLY and 0 nsec minimum after the the negation of the last RRPLY.

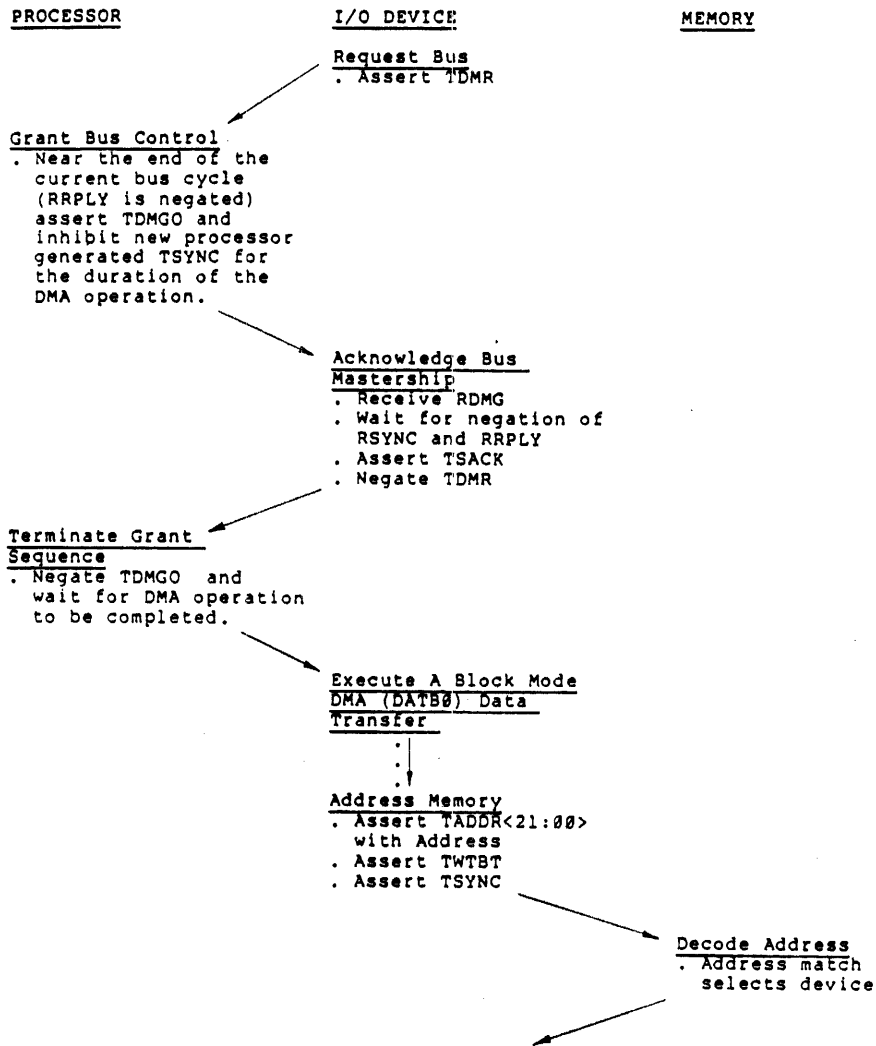
- o RELEASE THE BUS

- a) The DMA bus master negates TSACK 0 nsec after negation of the last RRPLY.
- b) The DMA bus master negates TSYNC 300 nsec maximum after it negates TSACK.
- c) The DMA bus master must remove TDATA, TBS7, and TWTBT from the bus 100 nsec maximum after clearing TSYNC.

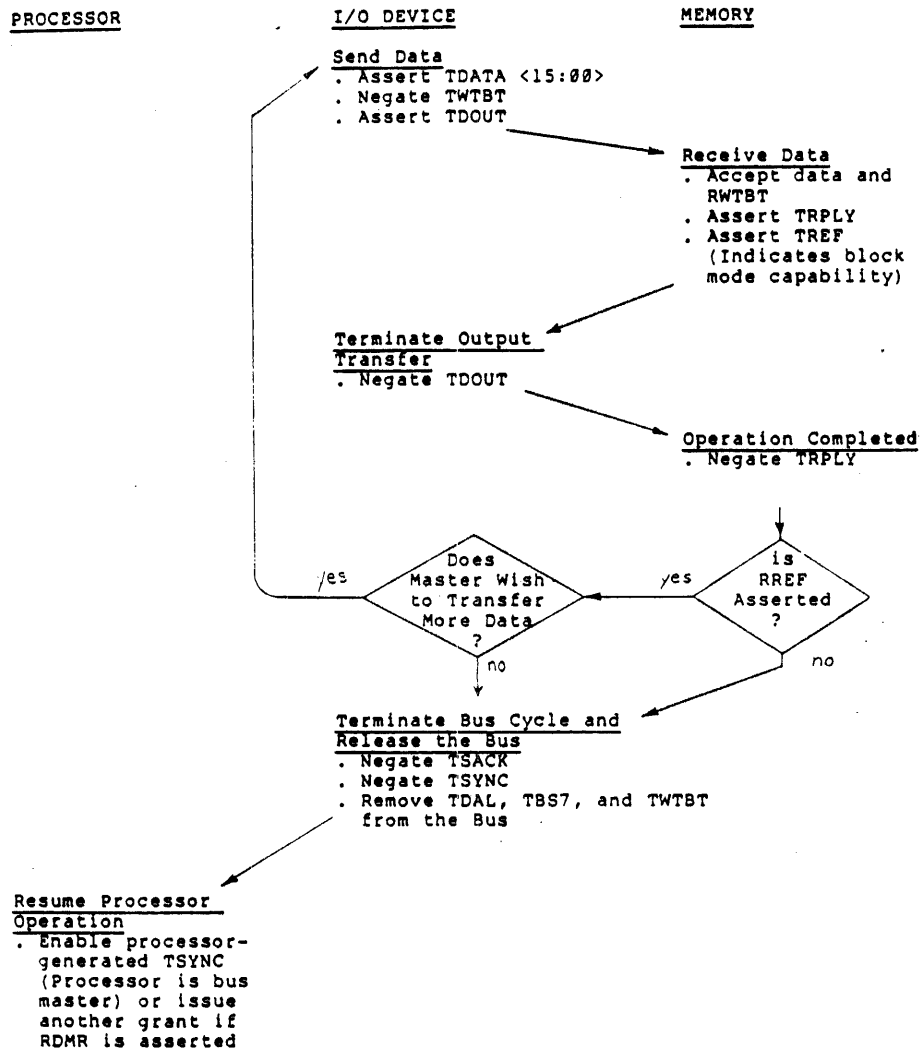
- o RESUME PROCESSOR OPERATION

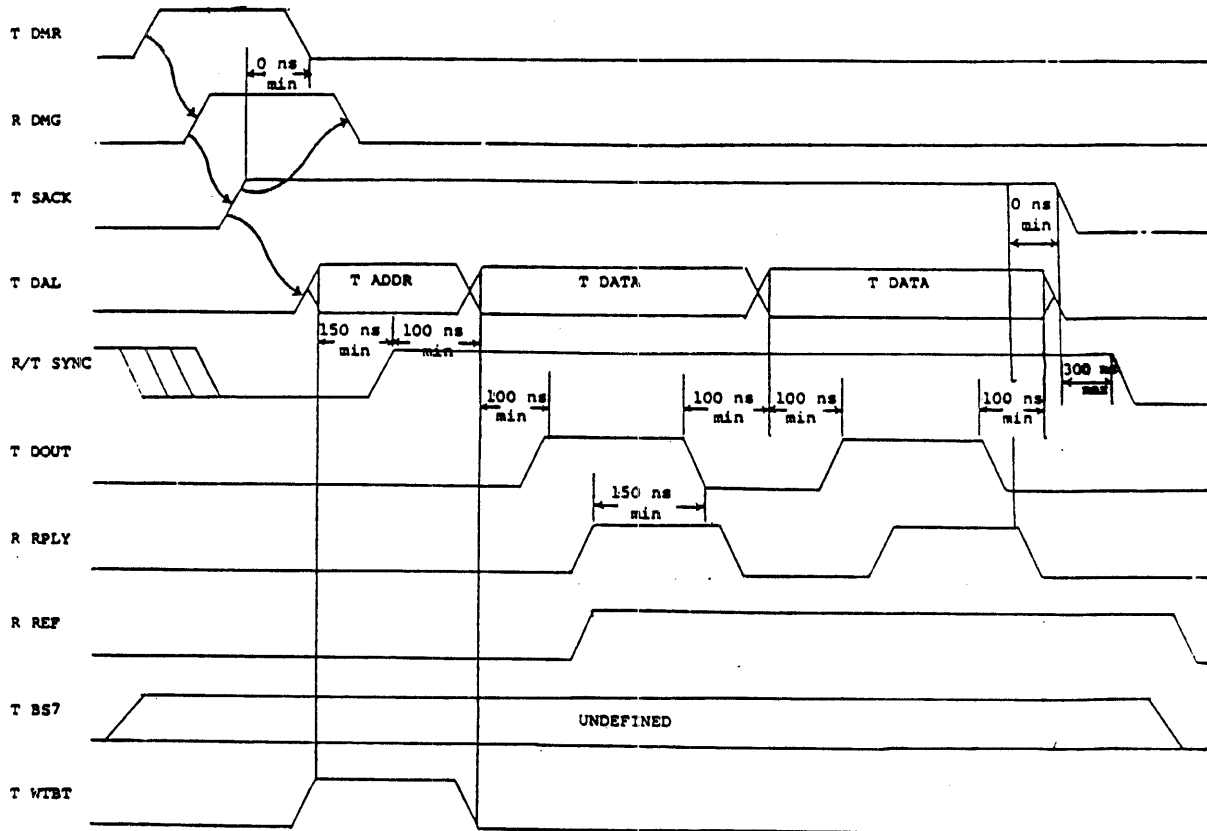
The bus arbitration logic in the CPU enables processor-generated TSYNC or will issue another bus grant (TDMGO) if RDMR is asserted.

## D A T B O C Y C L E



DATBO CYCLE CONT'D

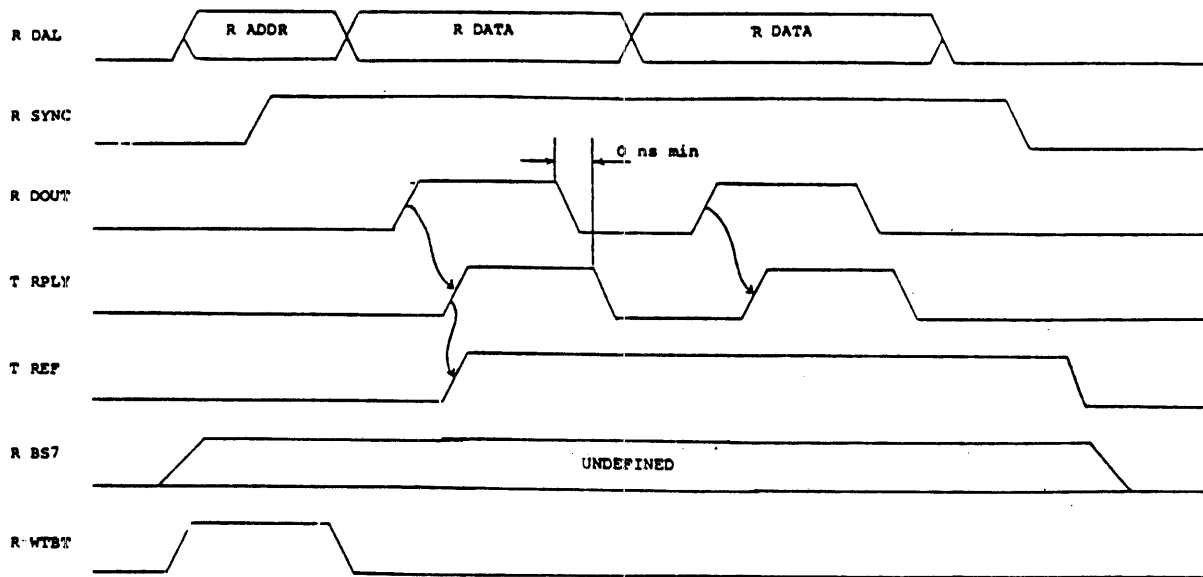




Timing at master device.  
T = Bus driver input  
R = Bus receiver output

DATBO

digital



Timing at slave device.  
T = Bus driver input  
R = Bus receiver output

DATBO

# uNOTE

NUMBER

114

DATE

11 / 28 / 83

TITLE Compatible Bootstrap for the LSI-11/73DISTRIBUTION UnrestrictedORIGINATOR Mike Collins

PRODUCT

LSI-11/73

PAGE 1 OF 3

The 11/73 (KDJ11-AA) is a high performance CPU for the QBus. It is a CPU only, which means that there is no boot capability on the module itself. Therefore a boot module must be selected to work with the 11/73.

This uNOTE will discuss the bootstrap modules which can be used with the 11/73.

There are 4 possible modules which can be used for bootstrap. They are: MXV11-BF w/MXV11-B2 boot ROMs  
MRV11-D w/MXV11-B2 boot ROMs  
MXV11-AA or -AC w/MXV11-A2 boot ROMs  
BDV11

For an 11/73 (KDJ11-AA) based system to be Field Serviceable the bootstrap code must execute a cache memory diagnostic on power-up. The only boot code which satisfies this requirement is found in the MXV11-B2 boot ROMs. Therefore an 11/73 based, Field Serviceable system must use either the MXV11-BF w/MXV11-B2 ROMs or the MRV11-D w/MXV11-B2 ROMs.

NOTE: The MXV11-B2 ROMs will not work on the MXV11-A module.

### MXV11-BF or MRV11-D w/MXV11-B2 ROMs

The MXV11-BF w/MXV11-B2 ROMs is the preferred choice since this module has 2 asynchronous serial lines as well as 128Kb of dynamic RAM in addition to the boot capability. However, if your application does not need the extra serial lines and RAM, an alternate choice would be the MRV11-D w/MXV11-B2 ROMs.

**digital**  
**MICROCOMPUTER**  
**PRODUCTS**  
**GROUP**

The MXV11-B2 ROMs will boot the following devices:

RL01/RL02 (DL)  
RX01/RX02 (DX,DY)  
TU58 (DD)  
TSV05 (MS)  
MSCP type devices e.g. RD51, RX50 (DU)  
DECnet via DPV11, DLV11-E, DLV11-F, DUV11

The remaining 2 boot modules do NOT have the necessary cache memory diagnostic code to make an 11/73 based system Field Serviceable.

Below is a list of all of the TESTED bootstraps for the 2 remaining boot modules.

MXV11-A w/MXV11-A2 ROMs.

Working Bootstraps:

RL01/RL02  
RX01/RX02  
TU58 conventional boot  
TU58 standalone boot

WARNING: If the MXV11-A is used in a 22 bit system the on-board RAM must be disabled. Refer to uNOTE #106

BDV11

Working Boostrops:

RL01/RL02  
RX02  
RK05

WARNING: Disable the processor and memory tests since an odd address trap does occur in each of them. (See NOTE below). To disable the CPU test, set switch E15-1 to OFF. To disable the memory test set switch E15-2 to OFF. Refer to the Microcomputer and Interfaces Handbook for complete configuration information.

The 11/73 has an on-board Line Time Clock Register, therefore the BDV11 BEVNT switch E21-5 should be set to the OFF position. This disables the BDV11 LTC reg. and allows the BEVNT signal to be under S/W control of the 11/73 LTC Reg.

If the BDV11 is used in a 22 bit system, it must be CS REV E or later or ECO M8012-ML0005 must be installed.

**NOTE:** ODD ADDRESS TRAPS. The 11/23 ignores an odd address reference whereas the KDJ11-A will trap to 4.



<b>uNOTE</b>		NUMBER 115
TITLE	LSI-11/73 Upgrade Paths	DATE 11 / 28 / 83
DISTRIBUTION	Unrestricted	PRODUCT LSI-11/73
ORIGINATOR	Mike Collins	PAGE 1 OF 7

### LSI-11/73 UPGRADE PATHS

With the announcement of the 11/73 (KDJ11-AA) CPU module, there will be numerous questions regarding configuring the module into a current system. The purpose of this uNOTE is to address all possible configuration upgrade paths (within reason).

Generally an 11/73 will be installed as an upgrade to a system built from components or a DEC packaged system.

In the case of a component upgrade it is assumed that the processor is a KDF11-A and the boot mechanism is an MXV11-A with the MXV11-A2 boot ROMs.

System upgrades fall into two categories:

1. KDF11-A Based Systems and
2. KDF11-B Based Systems (11/23+ and uPDP-11)

There are three issues which must be addressed when considering a KDJ11-A upgrade. They are:

1. The Boot Mechanism
2. 18 or 22 Bit System
3. Single or Multiple Box System

#### NOTE:

1. In the following upgrade scenarios, the systems have been labeled as being Field Serviceable or not. A system which is Field Serviceable has a bootstrap which meets Field Service requirements. The requirement is that the bootstrap must execute an 11/73 cache memory diagnostic on power-up. Reference uNOTE entitled "Compatible Bootstraps for the LSI-11/73". There is no guarantee that the overall system will be Field Serviceable or that it will be FCC compliant.

**digital**  
**MICROCOMPUTER**  
**PRODUCTS**  
**GROUP**

2. Systems using CPUs other than the KDF11-A or KDF11-B (i.e. 11/03 systems) are not considered for upgrade.

**CAUTION:**

It is recommended that the AC and DC loading for the final configuration be checked for conformance with the Q-bus loading rules. It is also recommended to check for overloading on the +5 Volt and +12 Volt Power Supplies.

For each system upgrade the following parameters are listed for both the "Current" system and the "Upgraded" system:

1. CPU
2. Boot Mechanism
3. System Size
4. Number of Boxes
5. Field Serviceable or Not
6. Special Conditions

**COMPONENT UPGRADE PATHS:**

1. Current System  
KDF11-A  
MXV11-A  
18 Bit System  
1 Box

Upgrade 1  
KDJ11-A  
MXV11-B/MRV11-D with MXV11-B2  
Boot ROMs  
18 Bit System  
1 Box  
Field Serviceable

Upgrade 2  
KDJ11-A  
MXV11-A  
18 Bit System  
1 Box  
NOT Field Serviceable

- |    |   |   |
|----|---|---|
| 2. | <u>Current System</u><br>KDF11-A<br>MXV11-A<br>18 Bit System<br>More than 1 Box   | <u>Upgrade</u><br>See Upgrades for #1   |
| 3. | <u>Current System</u><br>KDF11-A<br>MXV11-A (Memory Disabled)<br>22 Bit System<br>1 Box   | <u>Upgrade</u><br>See Upgrades for #1   |
| 4. | <u>Current System</u><br>KDF11-A<br>MXV11-A (Memory Disabled)<br>22 Bit System<br>More than 1 Box<br>This system is not currently | <u>Upgrade</u><br>Not currently configurable with<br>DEC equipment<br>configurable with DEC equipment |

#### PDP 11/23A SYSTEM UPGRADE PATHS:

- |    |   |   |
|----|---|---|
| 5. | <u>Current System</u><br>KDF11-A<br>BDV11<br>18 Bit System<br>1 Box | <u>Upgrade 1</u><br>KDJ11-A<br>MXV11-B/MRV11-D with MXV11-B2<br>Boot ROMs<br>18 Bit System<br>1 Box<br>Field Serviceable  |
|    |   | <u>Upgrade 2</u><br>KDJ11-A<br>BDV11<br>18 Bit System<br>1 Box<br>NOT Field Serviceable<br>Disable the Processor and<br>Memory tests and also the BEVNT<br>register on the BDV11. |

6. Current System  
KDF11-A  
BDV11  
18 Bit System  
More than 1 Box

Upgrade 3  
KDJ11-A  
MXV11-A (with MXV11-A2 boot ROMs)  
18 Bit System  
1 Box System  
NOT Field Serviceable  
Check AC loading since termination was removed when the BDV11 was removed from the system.

Upgrade 1  
KDJ11-A  
MXV11-B/MRV11-D with MXV11-B2 Boot ROMs  
18 Bit System  
More than 1 Box  
Field Serviceable  
Use BCV1A and BCV1B expansion cables.

Upgrade 2  
KDJ11-A  
BDV11  
18 Bit System  
More than 1 Box  
NOT Field Serviceable  
Disable the Processor and Memory tests and also the BEVNT register on the BDV11.  
Use BCV1B cable set between 1st and 2nd box and the BCV1A cable set between the 2nd and 3rd box.  
NOTE: If in a 3 box system the expansion cable set lengths must differ by 4 ft.

Upgrade 3

KDJ11-A  
MXV11-A (with MXV11-A2 boot  
ROMs)  
18 Bit System  
More than 1 Box  
NOT Field Serviceable  
Use BCV1A and BCV1B expansion  
cables.

7. Current System

KDF11-A

BDV11

22 Bit System

1 Box

Systems with this configuration were never shipped by DEC.

PDP 11/23 PLUS SYSTEM UPGRADE PATHS:

8. Current System

KDF11-B

Boot is on CPU

22 Bit System

1 Box System

Upgrade 1

KDJ11-A

MXV11-B/MRV11-D with MXV11-B2  
boot ROMs

22 Bit System

1 Box

Field Serviceable

Upgrade 2

KDJ11-A

MXV11-A (with MXV11-A2 boot  
ROMs)

22 Bit System

1 Box

NOT Field Serviceable

Must Disable RAM on MXV11-A

- |   |  |
|---|--|
| 9. <u>Current System</u><br>KDF11-B<br>Boot is on CPU<br>22 Bit System<br>More than 1 Box | <u>Upgrade 3</u><br>KDJ11-A<br>BDV11<br>22 Bit System<br>1 Box System<br>NOT Field Serviceable<br>Must have BDV11 ECO M8012-ML005 installed.<br>Disable the Processor and Memory tests and also the BEVNT register on the BDV11. |
|   | <u>Upgrade 1</u><br>Not currently configurable with DEC equipment.   |

#### MICRO PDP-11 SYSTEM UPGRADE PATHS:

- |   |   |
|---|---|
| 10. <u>Current System</u><br>Micro PDP-11<br>KDF11-BE<br>Boot is on CPU<br>22 Bit System<br>1 Box System    | <u>Upgrade</u><br>Same as 11/23+ rules, see upgrades for #8 |
| 11. <u>Current System</u><br>Micro PDP-11<br>KDF11-BE<br>Boot is on CPU<br>22 Bit System<br>More than 1 Box | <u>Upgrade</u><br>Same as 11/23+ rules, see upgrades for #9 |

NOTE: It is not currently possible to expand out of the uPDP-11 while maintaining FCC compliance.

11/23 PLUS and uPDP-11 system upgrades will require an EXTRA backplane slot to accommodate the additional boot module (i.e. MXV11-X or BDV11).

The KDF11-BA and KDF11-BE (CPUs for 11/23+ and uPDP-11) have two asynchronous serial lines in addition to the CPU and Boot ROMs. When the 11/73 is substituted for these CPUs the two serial lines must be replaced. Since the MXV11-BF has two serial lines it is the preferred choice.

## 11/23-S SYSTEM UPGRADE SOLUTIONS:

- |     |   |                                       |
|-----|---|---------------------------------------|
| 12. | <u>Current System</u><br>KDF11-BA<br>Boot is on CPU<br>18 Bit System<br>1 Box system    | <u>Upgrade</u><br>See upgrades for #5 |
| 13. | <u>Current System</u><br>KDF11-BA<br>Boot is on CPU<br>18 Bit System<br>More than 1 Box | <u>Upgrade</u><br>See upgrades for #6 |

NOTE: It is not currently possible to expand out of the 11/23-S while maintaining FCC compliance.

<b>ynote</b>		<b>NUMBER</b> R12
<b>TITLE</b> <u>Expanding into a Ball-SA Box</u>		<b>DATE</b> 8 / 10 / 82
<b>DISTRIBUTION</b> <u>Restricted</u>		<b>PRODUCT</b> BALL-SA
<b>ORIGINATOR</b> <u>Peter Kent</u>		<b>PAGE 1 OF 2</b>

Because of the availability of one type of box - specifically the BALL-SA it might be desirable to be able to expand from one BALL-SA box to another BALL-SA box. This particular arrangement was tested with one configuration to determine the workability of both boxes.

The master box contained the following: KDF11-B, MSV11-P, DZV11, and M9401 (expansion module and cable).

The expansion BALL-SA contained M9400-YE (expansion module and cable), M9400-YB (REV11 terminator board), and RLV12.

Both expansion modules were converted to 22 bit by using the 4 ground lines HH, KK, MM, and PP on J1. This was accomplished by desoldering J1 and cutting the etch foil (refer to drawing) under J1 for those four pins. Four wires were then added to those pins and connected to the unused fingers BCl, BD1, BE1, BF1. HH was connected to BCl, KK to BD1, MM to BE1, and PP to BF1. There are 2 potential problems with this arrangement:

- 1) By using 4 of the ground leads, some of the noise suppression may be compromised because normally every other wire on the cables are alternated with ground wires - here 4 BDAL lines are interspersed with other signal lines;
- 2) Some of the newer modules do not have any spare gold fingers.

The four BDAL lines (18-21) were not terminated on the REV 11 board. There are spare resistors on the REV11 resistor DIP for termination. These terminations would have to be wired to the four spare gold fingers BCl, BD1, BE1, and BF1. This could be done by wiring pins 2 and 3 on E6 to BCl and BD1 and cutting the etches going to AA1 and AB1 and wiring these to BE1 and BF1.

If a KDF11-A processor board is used, the first backplane must have an additional 240 ohm termination so that the total lumped backplane impedance is 120 ohm. An M9400-YE expansion module has the necessary termination on it and could be used in this case.

The jumper W1 on the H9276-B backplane (line time clock) was removed as well as the cable to J1 on the Bezel control card (to disable front panel switch functions) for expansion BALL-SA box.

**digital**  
**MICROCOMPUTER**  
**PRODUCTS**  
**GROUP**



The system was successfully booted with RT-11 version 4.0 and some functions were exercised, such as the clock, the editor, directory, and typed without difficulty.

This test of the above mentioned arrangement was by no means all inclusive and could not possibly attempt to cover all possible configurations in both backplanes.

You are looking at side 2  
(solder side).

FLIP CHIP<sup>®</sup>  
SIDE 1

SL

make cut here

TL

FIGURE 1

AA2

AV2

